RJ 10030 (90526) June 13, 1996
Computer Science

IBM Research Report

MDL Estimation for Small Sample Sizes and Its Application to Linear Regression

Byron E. Dom

K54/B2
IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099
e-mail: dom@almaden.ibm.com

IBM Research Division
Almaden · T.J. Watson · Tokyo · Zurich

MDL Estimation for Small Sample Sizes and Its Application to Linear Regression

Byron E. Dom

June 13, 1996

Abstract

In this paper we examine the problem of performing *Minimum Description Length* (MDL) based estimation for small sample sizes. By this we mean cases where the asymptotic ($n \to \infty$) forms are not appropriate. The path to our solution is embedded in a careful examination of the issue of *completeness* in MDL-based estimation. We develop specific MDL-based code-length formulas for use as objective functions in linear regression (fitting functions linear in the adjustable parameters). The problem we address is regression where we not only want to find the best fit for a given model order/complexity (e.g. polynomial degree), but also the best order. The accuracy of this fitting process will be assessed by how well the fitted function matches the "true" underlying function in simulation experiments. We succeed in obtaining several non-asymptotic expressions (code-length formulas) for the problem we consider and these results are compared with each other and with the "classic" asymptotic $\frac{k}{2}\log n$ MDL formula in fairly extensive simulation curve-fitting experiments. All the non-asymptotic formulas do better than $\frac{k}{2}\log n$.

Index Terms: Minimum Description Length principle, Stochastic Complexity, order estimation, model selection, statistical estimation, linear regression.

1 Introduction

In the most general terms this paper is about the problem of performing statistical estimation; fitting models to data. In many fields problems of this type arise that are ill-posed in the formal sense. One large class of such problems are those where it is desired to estimate the *complexity* or *order* (usually associated with the number of adjustable parameters) of a model where the actual desired criterion for optimality is how well the estimated model performs at predicting future results of the modeled (estimated) process or at estimating a "true" underlying model. These objectives are frequently equivalent. The latter is frequently used as a type of filtering wherein the predicted value (predicted by the model) corresponding to one of the measured data points, is believed to be more accurate than the original point.

To perform this estimation we are given a collection of data values, which in many problems are referred to as "training data". There are well known techniques (e.g. least-squares) for estimating the optimal values of these parameters when the model complexity is fixed. Blind application of the associated criteria to the case where model complexity is also to be estimated can lead to absurd results, however, that certainly do worse than many other possible models at predicting future results. This phenomenon has been referred to variously as "fitting the noise", "over fitting", and other things.

It is in this sense that such problems are ill-posed. Some objective function whose minimum corresponds closely to this desired property is needed. This is exactly where the strength of the *Minimum Description Length* (MDL) principle[14] lies. In this paper, we derive MDL-based objective functions for the problem of performing linear regression (e.g. polynomial curve fitting and fitting Fourier series to data). What is new and important about our results is that (as opposed to previous formulas) they are non-asymptotic, holding for very small sample sizes. We will verify this with results from simulation experiments.

In what follows we will move the exposition back and forth between the abstract and the concrete; between the general and the specific. The motivation for this is pedagogical. To put things on a somewhat concrete footing we start with a brief definition of the linear-regression problem. Following this, we summarize the Minimum Description Length principle for estimation. Next, we return to our regression problem with MDL in hand and discuss the application of MDL to this problem using the classic two-part encoding formula and its associated parameter encoding cost. In Section 4.1 we then begin a general examination of *completeness* in MDL-based estimation. [The term "completeness" has a certain coding-theoretic definition, discussed below and in Appendix A. When we use the term in this sense, it and related terms such as "incompleteness" will be *emphasized* with italics.] This examination includes a presentation of general formulas proposed by Rissanen[15] for *incompleteness* removal in two-part coding. In addition the examination discusses certain potential model-class properties that, when present, simplify the process of applying these general *incompleteness*-removal formulas to specific problems. These properties are present in the linear-regression problem we solve. At this point we are ready (in Section 5) to begin our detailed treatment of the problem of order estimation in linear regression using MDL.

We begin Section 5 by deriving code-length formulas for a *maximum-likelihood* code [Defined in Section 5.]for our linear-regression problem. After we have obtained a code-length formula this way, we proceed, in Section 5.2, to derive two formulas based on a uniform-prior-based encoding of the model parameters. In Section 6 we describe a *complete* formula derived by Nohre[12] for the problem of polynomial curve fitting. At this point, we have described five different code-length formulas to be used in performing model-order estimation in linear regression. We then perform, in Section 7, an experimental comparison of these five formulas on a set of five different polynomials of degrees: $\{0, 1, 2, 3, 4\}$ plus added Gaussian noise of different σ parameters. Finally, in Section 8, we draw conclusions based on the theoretical and experimental results of the preceeding sections and comment on possible future work.

2 Preliminaries

2.1 The Basic Problem: Linear Regression with Order Determination

The specific problem we will examine and solve (in Section (5)) is that of linear regression where we not only want to find the best fit for a given order, but also the best order. We are given a vector $\mathbf{y}$ (of length n) of data, which is a vector [a collection of data values organized as a vector for mathematical convenience] of values of a scalar-valued dependent variable y (i.e. $\mathbf{y} \equiv \{y_i\}$). There is also a corresponding vector $\mathbf{x}$ of values of the independent variable x ($\mathbf{x} \equiv \{x_i\}$). [Most of what we develop also applies to vector-valued x and y. Associated differences will be described.] We want to fit these points to a parametric function with the goodness of fit being measured by how well the fitted function predicts y values for x values not used in the fit or how well it does at estimating the true underlying model when this model is known as in simulation experiments. The model we assume for this data is a "true" underlying function of unknown order (number of adjustable parameters) plus uncorrelated zero-mean Gaussian "noise". In *linear regression* this function is linear in the associated adjustable parameters. Thus the assumed form of the data is:

$$y_i \;=\; \sum_{j=1}^{m} \alpha_j \phi_j(x_i) \;+\; \nu_i, \tag{1}$$

here $\{\phi_j(x)\}$ are a set of any linearly independent basis functions, $\{\alpha_j\}$ are adjustable coefficients and $\{\nu_i\}$ are assumed to be independent and identically distributed (i.i.d.) following a Gaussian distribution with zero mean and a standard deviation that will also be treated as an adjustable parameter. An example of possible basis functions are the following monomials, usually used in polynomial curve fitting: $\phi(x) \in \{1, x, x^2, \ldots, x^{m-1}\}$ also a Fourier series is a possibility: e.g. $\{\phi_k(x) = \cos(k\omega_0 x),\ k = 0, 1, 2, \ldots, m-1\}$, where ω_0 is the *fundamental* frequency of the series. This general model is discussed in more detail in Section 2.3.

If our goodness-of-fit measure were simply how well the estimated function fit the given points, this problem would be ill-posed because (in the case of polynomials for example) the best fit by our criterion will be a polynomial of order one less than the number of data points (the *Lagrange interpolating polynomial*) or higher. Our stated goodness-of-fit measure addresses this issue, but is obviously not applicable in practice, because one does not usually possess "future data" [An exception to this is where large amounts of data are available for estimation and can be divided into *training* and *test* sets. We are not considering this case by definition.] or the "true" model. Certain techniques such as *cross validation*[11, 5] attempt to simulate this but they are computationally very expensive. Thus we need some further "regularizing" principle to penalize higher order regression functions relative to those of lower order. We hope that this penalty will just offset the better fit obtained at the "true" order. For this purpose we use the *Minimum Description Length* principle (MDL).

2.2 The Minimum Description Length Principle (MDL)

The *Minimum Description Length Principle* (MDL) is a paradigm for doing statistical estimation, specifically model selection. MDL's strength is in determining model *structure* (a.k.a. *order* or *complexity*), a relevant example of which is polynomial degree in curve fitting. MDL has been discussed in detail by Rissanen[14]. We only summarize it briefly and we will not go into the associated philosophical issues here. The estimation problem it addresses is the following. We are given:

1. a vector $\mathbf{y}$ of n data values, $\{y_i | i = 1, 2, \ldots, n\}$. In most cases $\mathbf{y}$ will be considered values of a dependent variable and we will be given an associated vector $\mathbf{x}$ of n values of an independent

variable, x: $\{x_i | i = 1, 2, \ldots, n\}$. We will denote the set of possible $\mathbf{y}$ values by Λ which may be either finite or infinite in extent. In general the individual components (points) of which $\mathbf{y}$ and $\mathbf{x}$ are composed will be vector-valued. In most of what we describe here, however, the components of $\mathbf{y}$ will be assumed scalars. On the other hand, what we develop here holds equally for either scalar or vector-valued x_i. While we will sometimes treat $\mathbf{y}$ and $\mathbf{x}$ as real-valued for theoretical purposes, we assume that (as is always true in the real world) they are given to finite precision (quantized). The *quantization cells* (sets of real-valued points that all quantize to the same $\mathbf{y}$ value) correspond to these vectors are represented by $\delta\mathbf{y}$ and $\delta\mathbf{x}$, where $\delta\mathbf{y}, \delta\mathbf{x} \subset \mathbb{R}^n$.

2. a class Ξ of candidate models. Each model instance (element of Ξ) will be specified by the pair $(\theta^{(\xi)}, \xi)$ with which is associated a probability distribution $P(\mathbf{y}|\theta^{(\xi)}, \xi)$, where the *model structure* ξ is inherently discrete, but $\theta^{(\xi)}$ is usually assumed to be real-valued ($\theta^{(\xi)} \in \Gamma(\xi) \subset \mathbb{R}^{k(\xi)}$, where $k(\xi)$ is the number of components in $\theta^{(\xi)}$) from a theoretical point of view. The examples of ξ and $\theta^{(\xi)}$ that we will be considering are regression order (the number of expansion coefficients) and the coefficients themselves. In the specific example that we will use in our simulation experiments ξ and $\theta^{(\xi)}$ correspond to polynomial degree and the associated polynomial coefficients respectively. [Actually $\theta^{(\xi)}$ will also include the standard deviation, σ. The relationship of these parameters to a probability model of the form $P(\mathbf{y}|\theta^{(\xi)}, \xi)$ will become apparent.]

The MDL principle might be stated as follows. Given the data $(\mathbf{x}, \mathbf{y})$ and class Ξ of models, we will choose the model $(\theta^{(\xi)}, \xi) \in \Xi$ that allows the data to be encoded with the shortest code length. The coding scheme will be one that uses the model $P(\mathbf{y}|\theta^{(\xi)}, \xi)$ in the sense of Shannon[16] to construct a code with lengths that are approximately $-\log P(\mathbf{y}|\theta^{(\xi)}, \xi)$ bits, where "log" represents the base-two logarithm. In traditional Shannon coding we assume that $P(\mathbf{y}|\theta^{(\xi)}, \xi)$ is known *a priori* (i.e. not estimated), which gives us our code-length function. Algorithms such as Huffman's (see for example [17, Section 2.5; p. 39ff]) can be used to actually construct such codes, but we won't be concerned with this. Our estimation will be based on the associated length formulas. As mentioned, in MDL-based estimation we use models of the form: $P(\mathbf{y}|\theta^{(\xi)}, \xi)$. These models can, in some cases, be quite complicated. Frequently, we will use models involving continuous probability density functions $f(\mathbf{y})$ and using this, $P(\mathbf{y}) = \int_{\{\mathbf{w} \in \delta\mathbf{y}\}} f(\mathbf{w}) d\mathbf{w}$, which is usually approximated by: $P(\mathbf{y}) = f(\mathbf{y}) \cdot \|\delta\mathbf{y}\|$, where $\|\delta\mathbf{y}\|$ represents the content or "volume" of the data quantization cell $\delta\mathbf{y}$.

There is more than one way to use $P(\mathbf{y}|\theta^{(\xi)}, \xi)$ to encode $\mathbf{y}$. [For a discussion of these see [14].] To encode our data using the assumed model we will use a *two-part* scheme in which we first encode the model and then the data using the model. In this scheme the quantity minimized is:

$$L(\xi) \ + \ L(\theta^{(\xi)}, \xi) \ + \ L(\mathbf{y}|\theta^{(\xi)}, \xi), \tag{2}$$

where $L(\xi)$ is the code length for the model structure, but we will usually omit it because it has little or no effect in the problem we consider, because $L(\xi)$ is nearly constant over all values of ξ. [This isn't true for all problems, however; for a counter example see [6]], $L(\theta^{(\xi)}, \xi)$ is the length associated with some encoding scheme for the parameters, and $L(\mathbf{y}|\theta^{(\xi)}, \xi)$ is the code-length for the data $\mathbf{y}$ obtained in a scheme utilizing the model described by $(\theta^{(\xi)}, \xi)$. In the usual two-part coding $L(\mathbf{y}|\theta^{(\xi)}, \xi)$ is given by $-\log P(\mathbf{y}|\theta^{(\xi)}, \xi)$. The value usually used for $\theta^{(\xi)}$ is $\hat{\theta}^{(\xi)}$ - the *maximum-likelihood* (ML) estimate, which maximizes $P(\mathbf{y}|\theta^{(\xi)}, \xi)$ (and thus minimizes $-\log P(\mathbf{y}|\theta^{(\xi)}, \xi)$) for fixed $\mathbf{y}$ and ξ. This may not exactly minimize (2) but will be close to the minimizing value because $L(\theta^{(\xi)}, \xi)$ will almost always be a slowly varying function of $\theta^{(\xi)}$, whereas $P(\mathbf{y}|\theta^{(\xi)}, \xi)$, considered as a function of $\theta^{(\xi)}$, will tend to be sharply peaked about $\theta^{(\xi)} = \hat{\theta}^{(\xi)}$. In the course of our work (and its description) we will investigate the use of alternate forms for $L(\mathbf{y}|\theta^{(\xi)}, \xi)$. These will still be based on $P(\mathbf{y}|\theta^{(\xi)}, \xi)$.

The parameter code length $L(\theta^{(\xi)}, \xi)$ is frequently associated with the Bayesian notion of a *prior*[1] $\pi(\theta^{(\xi)})$; the two being related by

$$\pi(\theta^{(\xi)}) \quad = \quad 2^{-L(\theta^{(\xi)}, \xi)}, \tag{3}$$

which derives from the *Kraft inequality* (see [2, Section 5.5]). Rissanen[13] has derived what is now considered the "classical" form for this parameter code length based on an asymptotic optimal-precision analysis. That result is:

$$L(\theta^{(\xi)}, \xi) \quad \sim \quad \frac{k}{2} \log n, \tag{4}$$

which only holds for large n, but has frequently been used even for small n due to the lack of a computationally tractable alternative. In the work described here we will investigate the use of this formula and other, non-asymptotic formulas for the small-n case. In fact, it is overcoming the shortcomings of (4), at least in the case of linear regression, that is our primary intent here.

2.3 Application of MDL to Linear Regression

Putting things on a more concrete footing, we will now discuss the application of MDL to our linear-regression problem where the model is encoded via the following parameters: the model order, the regression coefficients α ($\alpha \equiv \{\alpha_i\}$). and the standard deviation σ of the Gaussian. We will return to the issue of encoding these parameters after we discuss the encoding of the data $\mathbf{y}$ using them. Using the Gaussian probability density associated with these parameters, we know that we can construct a code with approximately $-\log P(\mathbf{y}|\alpha, \sigma)$ bits where by $P(\mathbf{y}|\alpha, \sigma)$ we mean a discrete probability computed using the described Gaussian linear-regression model. As mentioned in the previous section, this latter point is significant. Encoding of this type must be performed on discrete "symbols", which we might think of as finite-precision representations of real numbers. We will represent the associated quantization step size by δy. [Some slight confusion related to notation is possible here. The quantity δy is a number: the quantization (precision) of the scalar data y. For our data vector $\mathbf{y}$, on the other hand, we let $\delta \mathbf{y}$ represent a region: the quantization cell corresponding to to the value $\mathbf{y}$.] While it is possible for every data point to have its own δy, we will assume that it is the same for all data points, which is the usual case. In fact, in many important practical applications the y values are integers ($\delta y = 1$) and the y values can always be rescaled such that this is the case. As we mentioned in the previous section, we assume that the underlying values are real numbers and distributed according to a continuous probability density

$f(y)$, which can be used to compute $P(y)$ as:

$$P(y) \quad = \quad \int_{y-(\delta y/2)}^{y+(\delta y/2)} f(w)dw, \tag{5}$$

which can usually be approximated by $f(y) \cdot \delta y$ and this is what we assume here. We should point out, however, that this will occasionally not be the case in practical applications and special measures will be required to deal with such situations. Since, for a given problem, δy (and therefore $\delta \mathbf{y}$) is fixed, it will be the same for all models and we will therefore omit the $\log ||\delta \mathbf{y}||$ term from most of our formulas.[$||\delta \mathbf{y}|| \equiv$ content or volume of $\delta \mathbf{y}$.] If we rescale the data values to integers by dividing by the precision δy, then $||\delta \mathbf{y}|| = 1$ which means that $\log ||\delta \mathbf{y}|| = 0$. Thus in most cases we take the data-encoding length to be $-\log f(\mathbf{y})$.

At this point, we will expand our notation to show the dependence on the parameters θ explicitly. In doing this we will replace ξ with m (the number of regression coefficients) and include $\mathbf{x}$ in the model description as a reminder that the polynomial coefficients must be used in conjunction with $\mathbf{x}$ to compute values of f. Thus the data code length will be written: $-\log f(\mathbf{y}|\alpha^{(m)}, m; \mathbf{x})$ where $\alpha^{(m)} \equiv \{\alpha_1, \alpha_2, \ldots, \alpha_m\}$,

$$f(\mathbf{y}|\alpha^{(m)}, m; \mathbf{x}) \quad = \quad \prod_{i=1}^{n} f(y_i|\alpha^{(m)}, m; x_i), \tag{6}$$

$$f(y|\alpha^{(m)}, m; x) \quad = \quad \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left[-\frac{1}{2}\left(\frac{y - \mu^{(m)}(x)}{\sigma}\right)^2\right], \tag{7}$$

and

$$\mu^{(m)}(x) \quad \equiv \quad \sum_{j=1}^{m} \alpha_j \phi_j(x). \tag{8}$$

In these expressions and in many of the subsequent formulas that appear $\theta = \{\sigma, \{\alpha_i | i = 1, 2, \ldots, m\}\}$.

Expanding $\log f$ for our assumed Gaussian gives:

$$-\log f(\mathbf{y}|\alpha^{(m)}, m; \mathbf{x}) \quad = \quad \frac{n}{2}\log 2\pi \;+\; n\log\sigma \;+\; \left(\frac{\log e}{2\sigma^2}\right)\sum_{i=1}^{n}(y_i - \mu^{(m)}(x_i))^2, \tag{9}$$

where the value of $\theta^{(m)} = (\alpha^{(m)}, \sigma)$ that minimizes (9) is the ML estimate $\hat{\alpha}(m)$ and the ML estimate for σ^2 is the sample variance:

$$\hat{\sigma}^2 \quad = \quad = \quad \frac{1}{n}\sum_{i=1}^{n}(y_i - \mu^{(m)}(x_i))^2. \tag{10}$$

[The normalizing term (n) in this expression is what one obtains in solving for the ML value of σ. This differs from the usual unbiased estimate of σ^2 where the normalizing factor is $(n - m)$.]

Substituting from equation (10) into (9) for σ yields:

$$-\log f(\mathbf{y}|\hat{\alpha}^{(m)}, \hat{\sigma}, m) \quad = \quad \frac{n}{2}\log(2\pi e) \;+\; n\log\hat{\sigma}, \tag{11}$$

where we have simplified the notation slightly by dropping the explicit dependence on $\mathbf{x}$.

Equation (11) is the encoding length for the data using the parameters $(\hat{\alpha}, \hat{\sigma})$ in a two-part scheme. For the decoder to be able to decode the data it must know these model parameters, which must, therefore, be encoded also. This model encoding cost is somewhat less straightforward to determine than the data encoding cost, but the approach we use will be crucial in determining the results we obtain. Using the classic MDL parameter encoding cost of equation (4) we obtain a total two-part encoding cost of:

$$L(\mathbf{y}) \;=\; \frac{n}{2}\log(2\pi e) \;+\; n\log\hat{\sigma} \;+\; \frac{m+1}{2}\log n, \tag{12}$$

which is both *incomplete* and asymptotic. This is one of the objective functions that we seek to improve upon. These encoded parameters $(\hat{\alpha}^{(m)}, \hat{\sigma})$, contain information about $\mathbf{y}$, however, because only certain $\mathbf{y}$ values could have led to these code-length-minimizing parameter values. This observation leads to the issue of *completeness* which will be key in our development.

3 Completeness in MDL Estimation

As has been pointed out (see for example [14, p.58]), the two-part coding scheme has the problem of being *incomplete* (defined/explained below). Nonetheless, because of their tractability, MDL estimation formulas based on it are the most frequently used (compared to those based on other, *complete* schemes) and the most frequently used among those use the asymptotic parameter code length of equation (4). The fact that these formulas are both *incomplete* and asymptotic may cause problems where ever small data sets (i.e. small n) must be used in doing estimation that includes estimating the model structure. A case in point is the problem of *image segmentation*, which has been treated using MDL methods[6, 10]. These segmentation algorithms perform a type of piecewise surface fitting where an image is divided into regions and a two-dimensional polynomial is fit to each region. The individual regions may be quite small (even one pixel) and accurate code length formulas become important not just for their utility in estimating the correct polynomial order, but also in making decisions like whether or not to merge adjacent regions.

The *completeness* issue is the following. In a real coding scheme, if the decoder knows the procedure that the encoder will use to estimate $(\theta^{(\xi)}, \xi) \rightarrow (\hat{\theta}^{(\hat{\xi})}, \hat{\xi})$, it can use this information to reduce the possible values of $\mathbf{y}$ once it "knows" $(\hat{\theta}^{(\hat{\xi})}, \hat{\xi})$ (i.e. $\mathbf{y}$ must be contained in the "inverse image" in $\mathbb{R}^n$ of $(\hat{\theta}^{(\hat{\xi})}, \hat{\xi})$, which we represent by $\Omega(\hat{\theta}^{(\hat{\xi})}, \hat{\xi})$.). For this reason the two-part scheme represented by (2) allows for possibilities that can never exist. That is, in the usual two-part coding, even though the encoder has encoded $(\hat{\theta}^{(\hat{\xi})}, \hat{\xi})$ and transmitted it to the decoder, it uses codes for $\mathbf{y}$ that assume that any value of $\mathbf{y}$ can occur in combination with $(\hat{\theta}^{(\hat{\xi})}, \hat{\xi})$, which obviously isn't the case. As a result, the probability $P(\mathbf{y}|\hat{\theta}^{(\hat{\xi})}, \hat{\xi})$ is normalized to sum to 1 when summed over all $\mathbf{y}$ values and the associated codebook would contain codes for all $\mathbf{y}$ values. A slightly more in-depth discussion of the issue of *completeness* is presented in Appendix A.

It is worth reminding the reader that one of the foundational philosophical tenets of MDL is that the code-length-based objective functions for doing model-order estimation should be based on an encoding scheme that gives the shortest possible code lengths when using models from the considered model class. For this reason, it is clearly desirable to ensure that the associated codes are *complete* where ever possible, and if doing so leads to non-asymptotic formulas, so much the better.

4 Removing Incompleteness in Two-Part Coding

4.1 General Incompleteness Removal in Two-Part Coding

A shorter code can thus be constructed by renormalizing $P(\mathbf{y}|\hat{\theta}^{(\hat{\xi})},\hat{\xi})$ to sum to 1 when summed over only those $\mathbf{y}$ values that could have resulted in the estimates $(\hat{\theta}^{(\hat{\xi})},\hat{\xi})$. Denote this set by $\Omega(\hat{\theta}^{(\hat{\xi})},\hat{\xi})$. Thus $L(\mathbf{y}|\theta^{(\xi)},\xi)$ in equation (2) can be replaced by:

$$L_{\mathrm{r}}(\mathbf{y}|\hat{\theta}^{(\hat{\xi})},\hat{\xi}) \;=\; -\log\left[\frac{P(\mathbf{y}|\hat{\theta}^{(\hat{\xi})},\hat{\xi})}{\sum_{\mathbf{w}\in\Omega(\hat{\theta}^{(\hat{\xi})},\hat{\xi})} P(\mathbf{w}|\hat{\theta}^{(\hat{\xi})},\hat{\xi})}\right], \tag{13}$$

where we've also replaced $(\theta^{(\xi)},\xi)$ by the code-length-minimizing values $(\hat{\theta}^{(\hat{\xi})}(\mathbf{y}),\hat{\xi}(\mathbf{y}))$. The subscript "c" on L_{c} in equation (13) stands for "complete".

If we have an encoding scheme for the parameters $\theta^{(\xi)}$ with an associated length function $L(\theta^{(\xi)},\xi)$, we can write our total code length as:

$$L_{\mathrm{c}}(\mathbf{y}) \;=\; L_{\mathrm{c}}(\mathbf{y}|\hat{\theta}^{(\hat{\xi})},\hat{\xi}) \;+\; L(\hat{\theta}^{(\hat{\xi})},\hat{\xi}). \tag{14}$$

It is interesting to note that:

$$\sum_{\mathbf{y}\in\Lambda} 2^{-L_{\mathrm{c}}(\mathbf{y})} \;=\; 1, \tag{15}$$

[where Λ is the set of all possible $\mathbf{y}$ values] in this case (i.e. we've succeeded in removing the *incompleteness*) and as a result, we can think of two-part coding with *incompleteness* removal as simply one-part (in the spirit of Shannon) coding with a somewhat complicated probability distribution. In this view the parameter estimation (i.e. in $\hat{\theta}(\mathbf{y})$) can be considered to be "under the covers".

4.2 The Sufficiency of Certain ML Estimators $\hat{\theta}(\mathbf{y})$

Many ML estimators $\hat{\theta}(\mathbf{y})$ have a type of sufficiency property where it is possible to manipulate $P(\mathbf{y}|\theta)$ into a new functional form $G(\hat{\theta}(\mathbf{y})|\theta)$ that is only a function of $\hat{\theta}(\mathbf{y})$. If this property holds, P can be taken outside the sum in the denominator of (13), cancelling with the numerator and collapsing the entire expression into $\log\|\Omega(\hat{\theta}^{(\hat{\xi})},\hat{\xi})\|$. That is, if we can write

$$P(\mathbf{y}|\theta) \;=\; G(\hat{\theta}(\mathbf{y})|\theta), \tag{16}$$

then

$$L_{\mathrm{c}}(\mathbf{y}|\hat{\theta}^{(\hat{\xi})},\hat{\xi}) \;=\; \log\left\|\Omega(\hat{\theta}^{(\hat{\xi})},\hat{\xi})\right\|, \tag{17}$$

where $\left\|\Omega(\hat{\theta}^{(\hat{\xi})},\hat{\xi})\right\|$ represents the number of data "states" (possible quantized values of $\mathbf{y}$) in $\Omega(\hat{\theta}^{(\hat{\xi})},\hat{\xi})$, which is the set of all such states whose corresponding parameter estimates, $\hat{\theta}^{(\hat{\xi})}$ are equal to that for our data vector $\mathbf{y}$. In other places we will use $\Omega(\hat{\theta}^{(\hat{\xi})},\hat{\xi})$ to represent a continuous region (manifold) containing these discrete states. In those cases the $\|\ldots\|$ notation represents the content of this manifold (in data space ($\mathbb{R}^n$)). This difference in the meaning of our notation should be clear from the context.

The result of equation (17) is significant in its simplicity. Also, as a practical matter, if we can obtain a closed-form expression for this quantity, it makes the estimation process much simpler. This property can easily be shown to hold for the Gaussian linear-regression models we consider here.

4.3 Incompleteness Removal Using an ML Code

In [15] Rissanen proposes addressing the *completeness* issue using a scheme based on the *maximum-likelihood* (ML) probability distribution $\hat{P}(\mathbf{y}|\xi)$:

$$\hat{P}(\mathbf{y}|\xi) \quad \equiv \quad \frac{P(\mathbf{y}|\hat{\theta}^{(\xi)}(\mathbf{y}), \xi)}{\sum_{\mathbf{w}\in\Lambda} P(\mathbf{w}|\hat{\theta}^{(\xi)}(\mathbf{w}), \xi)}, \tag{18}$$

where, the sum in the denominator is over all possible data vectors. Implicit in ξ is the number of components, k, in $\theta^{(\xi)}$ (i.e. $k = k(\xi)$). It should be clear that there is one such ML distribution for each model class and structure, ξ. [In the rest of this section, for notational simplicity, we will omit the explicit dependence of various quantities on ξ and k.]

In [15] Rissanen points out that using this form (equation (18)) is equivalent to *incompleteness* removal of the type of equation (14) with $L(\hat{\theta}^{(\hat{\xi})}, \hat{\xi}) = \log \pi_n(\hat{\theta})$ where:

$$\pi_n(\theta) \quad = \quad \frac{\sum_{\{\mathbf{y}|\hat{\theta}(\mathbf{y})=\theta\}} P(\mathbf{y}|\theta)}{\sum_{\{\mathbf{w}\in\Lambda\}} P(\mathbf{w}|\hat{\theta}(\mathbf{w}))}. \tag{19}$$

Using this expression for P, leads to the following approximation for (18) in terms of $\hat{f}$ (referred to as the *maximum-likelihood density*):

$$\hat{P}(\mathbf{y}) \quad \approx \quad \hat{f}(\mathbf{y}) \cdot \|\delta\mathbf{y}\| \quad = \quad \left[\frac{f(\mathbf{y}|\hat{\theta}(\mathbf{y}))}{\int_{\mathbf{w}\in\Lambda} f(\mathbf{w}|\hat{\theta}(\mathbf{w}))d\mathbf{w}}\right] \cdot \|\delta\mathbf{y}\|. \tag{20}$$

[A technical point: We have used Λ to represent both the region of integration (set of possible $\mathbf{w}$ values) for this integral and to represent the set of values over which the sum of equation (18) is taken. What we mean by Λ in the case of the integral is the obvious continuous region in $\mathbb{R}^n$ (implied by this discrete-to-continuous approximation) containing the discrete set Λ.]

For either of these (equations (18) and (20)) to be usable their denominators must (most importantly) remain finite and must also be possible to evaluate or at least approximate. The difficulty of achieving the latter of these requirements motivated Rissanen in [15] to derive the following approximate expression for the logarithm of the maximum-likelihood density (equation (20)):

$$-\log \hat{f}(\mathbf{y}) \quad = \quad -\log f(\mathbf{y}|\hat{\theta}(\mathbf{y})) \; + \; \frac{k}{2}\log\frac{n}{2\pi} \; + \; \log \int_{\Gamma} \sqrt{|I(\theta)|}d\theta \; + \; o(1), \tag{21}$$

where

- $I(\theta)$ is the *Fisher Information* matrix and $|I(\theta)|$ is its determinant. The $(i,j)^{th}$ component of $I(\theta)$ is:
$$-E_\theta \left\{ \frac{\partial \log f(x|\theta)}{\partial\theta_i \partial\theta_j} \right\}, \tag{22}$$
where $E_\theta\{\ldots\}$ denotes $\int\{\ldots\} f(x|\theta)d\theta$.

- $\Gamma \subset \mathbb{R}^k$ is the region of allowable θ values. Selection of Γ may be straightforward, being determined by the problem. It may (for example) be all of $\mathbb{R}^k$. In some cases, however, the integral $\int_\Gamma \sqrt{|I(\theta)|}d\theta$ may diverge when using the desired Γ. In such cases selection of an alternate Γ may be a delicate issue (refer to [15]).

This result, while still asymptotic, is much sharper than the previous $(\frac{k}{2}\log n)$ form.

Summarizing Rissanen's ML code-length functions, we are left with the possibility of using $L(\xi)$ plus one of: the log of (18), the log of (20), or (21), as the quantity to be minimized in MDL-based estimation, depending on the situation. Alternately, if we have another encoding scheme for the parameters, we can use it as indicated in equation (14). In what follows, we will not use the approximate form (21) but rather attempt to compute the ML probability ((18) or (20)) directly for the problem (linear regression) and model class we consider here.

4.4 Obtaining a Closed-Form Expression for $\hat{f}(\mathbf{y})$

In order to derive a usable code-length formula from the definition of $\hat{f}$ (equation (20)), it is desirable to evaluate it in closed form. Here we investigate potential properties of a model class that may make this possible.

It is possible to transform $\mathbf{y}$ as follows:
$$\mathbf{y} \rightarrow \mathbf{v} = (\mathbf{v}_1, \mathbf{v}_2) \tag{23}$$
where holding $\hat{\theta}(\mathbf{y})$ fixed is equivalent to fixing $\mathbf{v}_1$. The components of $\mathbf{v}_2$ are the coordinates specifying a point on the the corresponding manifold $\Omega(\hat{\theta})$. If our sufficiency property (Section 4.2) holds for the parameter estimates corresponding to $\mathbf{v}_1$, then we can write:
$$f(\mathbf{y}|\hat{\theta}(\mathbf{y})) \quad = \quad h(\mathbf{v}_1), \tag{24}$$
which, through the coordinate transformation of (23), serves as a definition of the function $h(\ldots)$. Using such a transformation, we may also write:
$$\hat{f}(\mathbf{y}) \quad = \quad \frac{h(\mathbf{v}_1)}{\int h(\mathbf{v}'_1) \int \left|\frac{\partial(\mathbf{y})}{\partial(\mathbf{v}')}\right| d\mathbf{v}'_1 d\mathbf{v}'_2} \tag{25}$$
where $|\partial(\mathbf{y})/\partial(\mathbf{v})|$ is the Jacobian determinant of the transformation of (23) and $\mathbf{v}' = (\mathbf{v}'_1, \mathbf{v}'_2)$ are dummy integration variables corresponding to $\mathbf{v} = (\mathbf{v}_1, \mathbf{v}_2)$. If this Jacobian further factors as:
$$\left|\frac{\partial(\mathbf{y})}{\partial(\mathbf{v})}\right| \quad = \quad j_1(\mathbf{v}_1) \cdot j_2(\mathbf{v}_2), \tag{26}$$
equation (25) can be written as:
$$\hat{f}(\mathbf{y}) \quad = \quad \frac{h(\mathbf{v}_1)}{\int h(\mathbf{v}'_1) j_1(\mathbf{v}'_1) \int j_2(\mathbf{v}'_2) d\mathbf{v}'_1 d\mathbf{v}'_2}. \tag{27}$$

Finally, if the $\mathbf{v}_2'$ limits are independent of $\mathbf{v}_1'$, we have

$$\hat{f}(\mathbf{y}) = \frac{h(\mathbf{v}_1)}{\left[\int h(\mathbf{v}_1')j_1(\mathbf{v}_1')d\mathbf{v}_1'\right]\left[\int j_2(\mathbf{v}_2')d\mathbf{v}_2'\right]}. \tag{28}$$

The significance of this form, which will be key in our analysis of linear regression, is that, if we obtain closed-form expressions for the integrals in the denominator, we obtain a closed-form expression for $\hat{f}(\mathbf{y})$. In the sequel we will see that our chosen Gaussian linear-regression models have all of the properties that make it possible to write $\hat{f}$ this way and that, further, we obtain integrals that we are able to evaluate.

5 Incompleteness Removal for Linear Regression With Gaussian Models

The goal of this section is to derive specific codelength formulas to be used as objective functions in doing linear regression with model-order estimation. In doing this we will initially treat all data and parameters as from a continuum (real numbers) as much as possible, defering discussion of the ultimately discrete (quantized) nature of these objects. Consistent with this intention, we will first derive formulas based on the ML code discussed in Section 4.3. Following that we will derive formulas based on an explicit coding of $\hat{\theta}^{(\hat{\xi})}$ using a certain uniform encoding scheme. We will divide the treatment of each of these problems into two parts. First, for primarily pedagogical reasons, we will derive the formula for fitting a constant (i.e. the sample mean) to three data points. We will then develop formulas for the more general case of n data points and m regression functions/coefficients, drawing analogies to the visualizable three-dimensional cases as much as possible.

5.1 Incompleteness Removal Using a Maximum-Likehood Code

5.1.1 A Simple Example: Three Data Points $(n = 3, m = 1)$

We begin this section with a discussion of the simple case $(n = 3, m = 1)$. We have three data points and we want to fit a constant to them. This constant is, in our case, the mean of a Gaussian distribution. It can be thought of as a 0^{th}-degree polynoimial in curve fitting, a 0^{th}-order Fourier series and so on. We assume, in this case, that the data values can be from a finite range of $[\mathsf{d}_{\min}, \mathsf{d}_{\max}]$ and we define $\mathsf{d} \equiv \mathsf{d}_{\max} - \mathsf{d}_{\min}$. We choose this case because it is easy to visualize. While rare, such cases do occur in practice. For example they are common in the early stages of the image segmentation algorithm described in [7]. These data points are scalar-valued and we will represent their values by: $\mathbf{y} = (x, y, z)$. [This x should not be confused with the independent variable.] Note that in this case Λ, the region of possible $\mathbf{y}$ values is a cube of side length d. We will refer to this cube in both three and n dimensions as the "data cube".

Thinking initially of the data as real-valued (i.e. not quantized), a surface of constant sample mean $\hat{\mu}$ in the space $(\mathbb{R}^3)$ corresponding to our three-dimensional example is the plane: $x + y + z = 3\hat{\mu}$. The "regression function" we fit in this case will be just the constant $\hat{\mu}$ [Specifying a 2nd-order (two regression coefficients) function would be equivalent to exactly specifying the data values

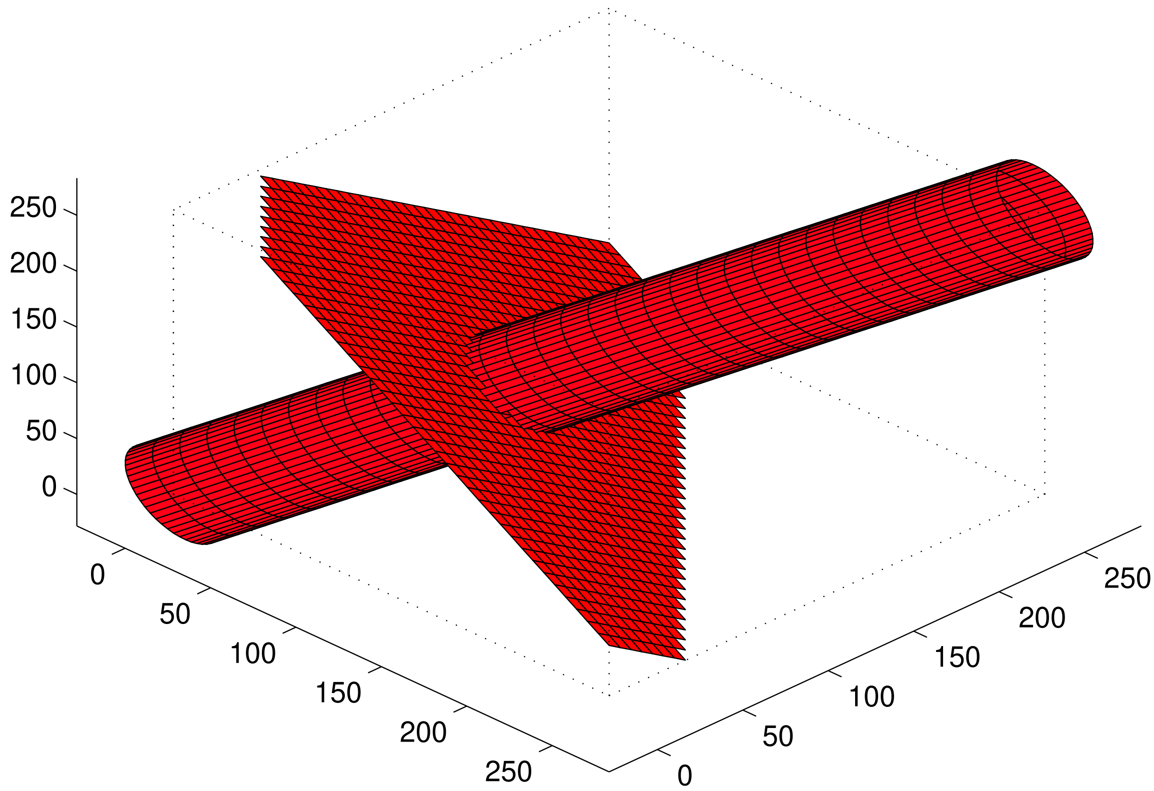


Figure 1: A visualization for the *constant-*$\hat{\theta}$ surfaces for the case of three data points. The cylinder is a surface of constant sample variance ($\hat{\sigma} = 20$) and the plane is a surface of constant sample mean ($\hat{\mu} = 100$). The dotted cube shows the region of possible data values $[\mathbf{d}_{\min}, \mathbf{d}_{\max}]$ which in the case (in the figure) is $[0, 256]$.

because it would require specifying three parameters - two regression coefficients and the sample standard deviation $\hat{\sigma}$.]. Thus, once we specify the estimated sample mean $\hat{\mu}$ we know that the possible data vectors $\mathbf{y}$ correspond to points lying in that plane.

A surface of constant sample variance $\hat{\sigma}^2$ is a cylinder of radius $\sqrt{3}\hat{\sigma}$ whose axis is the line $x = y = z$ (see Appendix C). Thus, when both of these parameters ($\hat{\mu}$ and $\hat{\sigma}$) have been specified, we know that the possible $\mathbf{y}$ values (in our infinite-precision example) lie on the intersection of the two corresponding surfaces (the plane and the cylinder). This intersection is a circle imbeded in $\mathbb{R}^3$. This three-point case is depicted schematically in Figure 1. The dotted lines in the figure outline Λ the *data cube*.

Now we begin the process of deriving an expression for $\hat{f}(\mathbf{y})$ (equation (20)). In keeping with our discussion of coordinate transformations in Section 4.4, we will perform two transformations in succession. These could be carried out as a single transformation, but doing them one at a time adds some insight. First we perform a simple rotation of the coordinate axes. We will symbolize this by:

$$(x, y, z) \quad \rightarrow \quad (u, v, w), \tag{29}$$

where $w = \sqrt{3}\hat{\mu}$ (i.e. the w axis is the line $x = y = z$.) and the (u, v) plane is normal to the w axis and goes through the origin. Because this transformation is *orthogonal* we have:

$$\left|\frac{\partial(x,y,z)}{\partial(u,v,w)}\right| \;=\; 1, \tag{30}$$

Relating this to Figure 1, the w axis is the axis of the cylinder and the (u,v) plane is the plane parallel to the plane in the figure but containing the origin.

Our second transformation is from the rectangular (u,v,w) system to a cylindrical-polar system (ρ,β,ζ):

$$(u,v,w) \;\rightarrow\; (\rho,\beta,\zeta), \tag{31}$$

where $\zeta = w$, $\rho = \sqrt{u^2+v^2}$ and $\beta = \arctan(v/u)$. This transformation, which is not orthogonal, has the Jacobian:

$$\left|\frac{\partial(u,v,w)}{\partial(\rho,\beta,\zeta)}\right| \;=\; \rho. \tag{32}$$

The Jacobian of the combined $((x,y,z)\rightarrow(\rho,\beta,\zeta))$ transformation is just the product of (30) and (32), so we have:

$$\left|\frac{\partial(x,y,z)}{\partial(\rho,\beta,\zeta)}\right| \;=\; \rho. \tag{33}$$

Relating this once again to Figure 1, ρ is the raidus of the cylinder shown there.

Next let us examine the function $f(\mathbf{y}|\hat{\theta}(\mathbf{y}))$ for our assumed Gaussian models. Manipulating equation (11) slightly gives:

$$f(\mathbf{y}|\hat{\theta}(\mathbf{y})) \;=\; \left[2\pi e\hat{\sigma}^2(\mathbf{y})\right]^{-\frac{3}{2}}. \tag{34}$$

From this, it is clear that the estimator $\hat{\sigma}(\mathbf{y})$ has the sufficiency property discussed in Section 4.2. Note also that in this case $\rho = \sqrt{3}\,\hat{\sigma}$. Thus we can write:

$$f(\mathbf{y}|\hat{\theta}(\mathbf{y})) \;=\; g(\rho) \;=\; \left(\frac{3}{2\pi e\rho^2}\right)^{\frac{3}{2}}, \tag{35}$$

which gives the following for the integral in the denominator of our expression for $\hat{f}$ (equation (20)).

$$\int f(\mathbf{y}|\hat{\theta}(\mathbf{y}))d\mathbf{y} \;=\; \int g(\rho)\rho d\rho \int d\beta \int d\zeta. \tag{36}$$

Substituting for $g(\rho)$ gives:

$$\int f(\mathbf{y}|\hat{\theta}^{(\xi)}(\mathbf{y}))d\mathbf{y} \;=\; \left(\frac{3}{2\pi e}\right)^{\frac{3}{2}} \int \rho^{-2}d\rho \int d\beta \int d\zeta. \tag{37}$$

To go further with this we need limits for the β, ρ and ζ integrals. For our limited-data-values case the integral we are evaluating is over Λ, which is a cube of side length d. This results in messy limits. A reasonable approximate alternative would seem to be the assumption that Λ is the bounding (circumscribed) sphere of our "data cube". This sphere will have diameter $\sqrt{3}\,\mathsf{d}$. From this point forward, we will redefine ζ so that it runs from 0 to $\sqrt{3}\,\mathsf{d}$. This is equivalent to letting $\mathsf{d}_{\mathsf{min}} = 0$ and is done to simplify the notation, but without loss of generality in the results.

In this case $\int d\beta = 2\pi$ and $\int d\zeta \rightarrow \int_0^{\sqrt{3}\,\mathsf{d}} d\zeta$. The ρ integral has a variable upper limit that depends on ζ however: $\rho_{\mathsf{max}}(\zeta) = \sqrt{\zeta(\sqrt{3}\mathsf{d}-\zeta)}$. [We could just as well make the ζ integral have variable (ρ-dependent) limits.] To simplify the mathematics we will simply assume that $\rho_{\mathsf{max}} =$

$\sqrt{3}\,(\mathsf{d}/2)$ (the radius of our assumed sphere) for all values of $\zeta \in [0, \sqrt{3}\,\mathsf{d}_{\max}]$. This assumption implies that Λ is actually a circular cylinder whose axis is the ζ axis and with height and diameter of the base both equal to $\sqrt{3}\,\mathsf{d}$, which gives us:

$$\int d\zeta \;\rightarrow\; \int_0^{\sqrt{3}\,\mathsf{d}} d\zeta \;=\; \sqrt{3}\,\mathsf{d}. \tag{38}$$

The ρ integral causes a problem, however, because it appears to include the singularity $\rho = 0$. This is an example of where the discrete-to-continuous approximation that took us from (18) to (20) breaks down. That is, we've been using the following approximation:

$$P(\mathbf{y}) \;=\; \int_{\mathbf{w}\in\delta\mathbf{y}} f(\mathbf{w})d\mathbf{w} \;\approx\; f(\mathbf{y})\cdot\|\delta\mathbf{y}\|. \tag{39}$$

As $\rho \to 0$ (i.e. $\hat{\sigma} \to 0$) the approximation breaks down and we must compute:

$$P(\mathbf{y}) \;=\; \int_{\mathbf{w}\in\delta\mathbf{y}} f(\mathbf{w})d\mathbf{w}. \tag{40}$$

We intend to examine this issue in detail in a future paper, but for now we will simply set a lower limit $\hat{\sigma}_0$ on $\hat{\sigma}(\mathbf{y})$ and define the problem to exclude these $\mathbf{y}$ values. [As a practical matter $\hat{\sigma}_0$ will be on the order of the data quantization and in the experiments whose results we present in the sequel $\hat{\sigma}_0$ was set equal to this quantization.] The ρ integral thus becomes:

$$\int_{\rho_0}^{\rho_{\max}} \rho^{-2}d\rho \;=\; \left(\frac{1}{\rho_0} - \frac{1}{\rho_{\max}}\right), \tag{41}$$

where $\rho_0 = \sqrt{3}\,\hat{\sigma}_0$, $\rho_{\max} = \sqrt{3}\,(\mathsf{d}/2)$. For our *unlimited-data-values* case $\rho_{\max} \to \infty$ so that the ρ integral becomes $1/(3\rho_0)$. In this case, the ζ integral gives us problems, however. It diverges. This problem and how to deal with it will be discussed in subsequent sections.

Returning to our original intention to obtain an expression for $\hat{f}$, which is defined as [repeating equation (18)]:

$$\hat{f}(\mathbf{y}) \;\equiv\; \frac{f(\mathbf{y}|\hat{\theta}(\mathbf{y}))}{\int_{\mathbf{w}\in\Lambda} f(\mathbf{w}|\hat{\theta}(\mathbf{w}))d\mathbf{w}}, \tag{42}$$

We have obtained the following results, which may now be applied to obtain the desired expression.

$$\begin{aligned} f(\mathbf{y}|\hat{\theta}(\mathbf{y})) &= \left[2\pi e\hat{\sigma}^2(\mathbf{y})\right]^{-\frac{3}{2}} \\ \int f(\mathbf{y}|\hat{\theta}(\mathbf{y}))d\mathbf{y} &= \int g(\rho)\rho d\rho \int d\beta \int d\zeta \\ \int d\beta \;=\; 2\pi; &\quad \int d\zeta \;=\; \sqrt{3}\,\mathsf{d} \\ \int g(\rho)\rho d\rho &= \left(\tfrac{3}{2\pi e}\right)^{\frac{3}{2}} \tfrac{1}{\sqrt{3}} \left(\tfrac{1}{\hat{\sigma}_0} - \tfrac{1}{(\mathsf{d}/2)}\right) \end{aligned} \tag{43}$$

Applying these yields:

$$\hat{f}(\mathbf{y}) \;=\; \frac{\hat{\sigma}(\mathbf{y})^{-3}}{3^{\frac{3}{2}}\left(\frac{1}{\hat{\sigma}_0} - \frac{1}{\hat{\sigma}_{\max}}\right)}, \tag{44}$$

where $\hat{\sigma}_{\max} = \mathsf{d}/2$.

5.1.2 The General Case: n Data Points and m Regression Functions

Having solved our problem for the simple case of fitting a constant to three data points, we are ready to tackle the more difficult general problem of fitting a sum of m linearly independent basis functions to a set of n data points. This is the general problem of linear regression. Once again, what we are after is the corresponding code-length formula, which is based on our assumed Gaussian linear-regression model and is a function of $m, n, \mathbf{x}$, and $\mathbf{y}$. This formula then provides an objective function to be used in determining model order, which means (in this case) the value of m. As in the three-dimensional case, this problem is essentially that of finding an expression for $\hat{f}$, whose general form is:

$$\hat{f}(\mathbf{y}) = \frac{f(\mathbf{y}|\hat{\theta}(\mathbf{y}))}{\int_{\mathbf{w}\in\Lambda} f(\mathbf{w}|\hat{\theta}(\mathbf{w}))d\mathbf{w}}. \tag{45}$$

As we showed in Section 2.3 the numerator of this expression can, for our linear-regression models, be reduced to:

$$f(\mathbf{y}|\hat{\theta}(\mathbf{y})) = \left[2\pi e\hat{\sigma}^2(\mathbf{y})\right]^{-\frac{n}{2}}, \tag{46}$$

which leaves us with the problem of finding an expression for the denominator of (45). As we did in the three-dimensional case, we will follow the program outlined in Section 4.4, which involves finding the right coordinate transformation.

Note that, if we think of our three-dimensional case as linear regression, then we were fitting $\mathbf{y}$ to a single regression function $\phi(\mathbf{x}) = 1$ [Could be any constant.] and our lone regression coefficient is $\alpha = \hat{\mu}$ (the sample mean). Note also that in this case our coefficient is obtained by doing what is equivalent to projecting the vector $\mathbf{y}$ (in $\mathbb{R}^3$) onto the ζ axis (which is also the w axis and the line $(x = y = z)$). It is well known in linear algebra[18] that linear regression can be seen and carried out as a projection operation. This point is key to the following discussion.

We will continue this discussion in the language of linear algebra and vector spaces. In that spirit we define the $n \times m$ matrix $\Phi^{(m)}(\mathbf{x})$ composed of $\{\phi_{ij} = \phi_j(x_i) \,|\, i = 1,\ldots,n;\ j = 1,\ldots,m\}$. When we write Φ without the superscript "(m)" it implies the case $m = n$. That is, the number of regression functions is equal to the number of data points. [In the case of polynomial curve fitting the elements of Φ are usually defined as $\phi_{ij} = x_i^{j-1}$ and Φ is referred to as *Vandermonde's* matrix.] If we have a set of n linearly independent basis functions and n unique values in $\mathbf{x}$, we can write:

$$\mathbf{y} = \Phi\,\alpha = \sum_{i=1}^{n} \phi_i \alpha_i \tag{47}$$

(where (in this context) $\{\phi_i\}$ are the columns of Φ) and expect to find an exact solution. If we have either less than n independent basis functions or less than n unique x values, then we can only find a least-squares solution.

Assume, for the moment, that we want the (m^{th}-order least-squares) solution to:

$$\mathbf{y} = \Phi^{(m)}\,\alpha^{(m)}. \tag{48}$$

The well known solution to this can be expressed as the projection of $\mathbf{y}$ onto the subspace of $\mathbb{R}^n$ spanned by the columns of $\Phi^{(m)}$, [This subspace is denoted by $R\left(\Phi^{(m)}\right)$ and referred to as the *column space* or *range* of $\Phi^{(m)}$.] which is given by[18]:

$$\hat{\alpha}^{(m)} = \left[\left(\Phi^{(m)}\right)^T \Phi^{(m)}\right]^{-1} \left(\Phi^{(m)}\right)^T \mathbf{y}. \tag{49}$$

There are other ways of expressing and computing this solution and some are numerically more desirable. This issue will not concern us at this point, however. We will write our least-squares fit in terms of this solution as:

$$\mu^{(m)} = \Phi^{(m)} \hat{\alpha}^{(m)}. \tag{50}$$

If we let $\nu^{(m)}$ represent the length-n vector of residuals to this fit, i.e.:

$$\mathbf{y} = \mu^{(m)} + \nu^{(m)}, \tag{51}$$

then it can be expressed and computed as the projection of $\mathbf{y}$ onto the *orthogonal complement* of $R\left(\Phi^{(m)}\right)$, which we represent by $R(\Phi^{(m)})^{\perp}$. Returning to our three-dimensional example, $R\left(\Phi^{(m)}\right)$ corresponds to the $w = \zeta$ axis and $R(\Phi^{(m)})^{\perp}$ corresponds to the (u, v) plane (the plane normal to the ζ axis and containing the origin).

Next we perform a coordinate transformation $(\mathbf{y} \to \eta)$ that is the analog of the $[(x, y, z) \to (u, v, w)]$ transformation we used in our three-dimensional example. In this case the first m components of η express the projection $\mu^{(m)}$ in terms of some orthonormal basis for $R\left(\Phi^{(m)}\right)$ and the last $(n - m)$ components express the residuals $\nu^{(m)}$ in terms of some orthonormal basis for $R\left(\Phi^{(m)}\right)^{\perp}$. If we construct an $n \times n$ matrix Ψ whose columns are these two sets of basis vectors, we can write:

$$\mathbf{y} = \Psi\, \eta \tag{52}$$

$$\mu^{(m)} = \Psi^{(m)}\, \eta^{(m)} \tag{53}$$

$$\nu^{(m)} = \Psi^{(m,n)}\, \eta^{(m,n)}, \tag{54}$$

where $\eta^{(m,n)}$ denotes the last $(n - m)$ components of η and $\Psi^{(m,n)}$ is an $n \times (n - m)$ matrix whose columns are our orthonormal basis for $R\left(\Phi^{(m)}\right)^{\perp}$.

As we did in our three-dimensional example we will follow this orthonormal transformation by a second transformation, which leaves the coordinates $\eta^{(m)}$ corresponding to $R\left(\Phi^{(m)}\right)$ unchanged, but expresses the coordinates $\eta^{(m,n)}$ corresponding to $R\left(\Phi^{(m)}\right)^{\perp}$ in the spherical-polar form: (ρ, Θ) where Θ consists of $(n - m - 1)$ angular coordinates $\{\beta_i\}$ and ρ is:

$$\rho = \|\eta^{(m,n)}\| = \|\nu^{(m)}\| = \sqrt{n}\, \hat{\sigma}. \tag{55}$$

Thus, for fixed $\eta^{(m)}$ (fixed $\alpha^{(m)}$) and fixed $\hat{\sigma}$, the manifold of possible $\eta^{(m,n)}$ values (in $\mathbb{R}^{n-m}$)) is the surface of a sphere of radius $\rho = \sqrt{n}\,\hat{\sigma}$ centered at the origin. The corresponding subset (in $\mathbb{R}^n$) of possible $\mathbf{y}$ values is the same sphere imbeded in the plane (hyperplane) that is the manifold of $\mathbf{y}$ values that would give the same $\eta^{(m)}$. That is the set of $\mathbf{y}$ values whose projection onto $R\left(\Phi^{(m)}\right)$ is equal to $\mu^{(m)}$. [Remember: $\mathbf{y}, \mu^{(m)} \in \mathbb{R}^n$ while $\eta^{(m)} \in \mathbb{R}^m$ and $\eta^{(m,n)} \in \mathbb{R}^{n-m}$.]

The Jacobian determinant for our second transformation is[8]:

$$\left|\frac{\partial\left(\eta^{(m,n)}\right)}{\partial(\rho, \Theta)}\right| = \rho^{n-m-1} c_1^{n-m-2} c_2^{n-m-3} \ldots c_{n-m-2} \equiv \rho^{n-m-1} C(\Theta), \tag{56}$$

where $c_i \equiv \cos\beta_i$, etc. [This equation serves as the definition of $C(\Theta)$.] Note that this is a product of terms, each of which is a function of only one coordinate. This is the kind of separability we were looking for in Section 4.4.

Once again, as in our three-dimensional example, we will take advantage of the sufficiency of $\hat{\sigma}(\mathbf{y})$, which allows us to write:

$$f(\mathbf{y}|\hat{\theta}(\mathbf{y})) \quad = \quad g(\rho; n) \quad = \quad \left(\frac{n}{2\pi e \rho^2}\right)^{\frac{n}{2}}. \tag{57}$$

Using this and the Jacobian (equation (56)) we can write:

$$\int f(\mathbf{y}|\hat{\theta}^{(\xi)}(\mathbf{y}))d\mathbf{y} \quad = \quad \int \rho^{n-m-1} g(\rho; n) d\rho \int C(\Theta) d\Theta \int d\eta^{(m)} \tag{58}$$

Equation (36) is the $(n = 3, m = 1)$ version of this. Substituting for $g(\rho; n)$ gives:

$$\int f(\mathbf{y}|\hat{\theta}^{(\xi)}(\mathbf{y}))d\mathbf{y} \quad = \quad \left(\frac{n}{2\pi e}\right)^{\frac{n}{2}} \int \rho^{-(m+1)} d\rho \int d\Theta \int d\eta^{(m)}. \tag{59}$$

For limited data values the "exact" region of integration Λ is an n-dimensional cube of side length d, which we will approximate by a region analogous to the cylinder we used in our three-dimensional example. Evaluating the integrals in equation (59), using the limits corresponding to our approximate Λ we obtain[8] first:

$$\int C(\Theta) d\Theta \quad = \quad \frac{2\pi^{(n-m)/2}}{\Gamma((n-m)/2)}, \tag{60}$$

which is $(n-m)$-dimensional generalization of the total solid angle for $\mathbb{R}^{(n-m)}$ and, when multiplied by $\rho^{(n-m-1)}$, gives the content ("surface area") of the surface of an $(n-m)$-dimensional sphere of radius ρ[8]. Next we evaluate the ρ integral:

$$\int_{\rho_0}^{\rho_{\max}} \rho^{-(m+1)} d\rho \; = \; \frac{1}{m} \left(\frac{1}{\rho_0^m} \; - \; \frac{1}{\rho_{\max}^m}\right), \tag{61}$$

where, analogous to the three-dimensional case, $\rho_0 = \sqrt{n}\,\hat{\sigma}_0$ and $\rho_{\max} = \sqrt{n}\,\mathsf{d}/2$. [This is analogous to our cylinder assumption for Λ in the three-dimensional case.] This brings us to the problem of evaluating $\int d\eta^{(m)}$.

The possible values of $\eta^{(m)}$ are those in the projection of Λ onto $R\left(\Phi^{(m)}\right)$. The content of this projection is $\int d\eta^{(m)}$. Continuuing with our approximation of Λ, the region of possible $\eta^{(m)}$ values is then the projection of this sphere onto $R\left(\Phi^{(m)}\right)$. The content of this projection is that of an m-dimensional sphere of radius $(\sqrt{n}\,\mathsf{d})/2$ (half the diagonal of the data cube). The expression for this is[8]:

$$\int d\eta^{(m)} \quad = \quad \frac{2\pi^{\frac{m}{2}} n^{\frac{m}{2}} (\mathsf{d}/2)^m}{m\Gamma(\frac{m}{2})}. \tag{62}$$

Combining our evaluations of these three integrals $(\Theta, \rho$ and $\eta^{(m)})$ gives:

$$\int_\Lambda f(\mathbf{y}|\hat{\theta}^{(\xi)}(\mathbf{y}))d\mathbf{y} \quad = \quad \left(\frac{n}{2\pi e}\right)^{\frac{n}{2}} \left[\frac{2\pi^{(n-m)/2}}{\Gamma((n-m)/2)}\right] \left[\frac{1}{mn^{\frac{m}{2}}}\left(\frac{1}{\hat{\sigma}_0^m} \; - \; \frac{1}{(\mathsf{d}/2)^m}\right)\right] \left[\frac{2\pi^{\frac{m}{2}} n^{\frac{m}{2}} (\mathsf{d}/2)^m}{m\Gamma(\frac{m}{2})}\right]. \tag{63}$$

Simplifying and factoring out the n-dependent terms gives:

$$\int_\Lambda f(\mathbf{y}|\hat{\theta}^{(\xi)}(\mathbf{y}))dy \;=\; \left[\frac{n^{n/2}}{(2e)^{\frac{n}{2}}\Gamma((n-m)/2)}\right]\left\{\frac{4}{m^2\Gamma\left(\frac{m}{2}\right)}\left[\left(\frac{\mathsf{d}}{2\hat{\sigma}_0}\right)^m - 1\right]\right\} \tag{64}$$

[In Appendix B we show that this goes asymptotically ($n \to \infty$) to $\frac{m+1}{2}\log n$, which agrees with Rissanen's classic "$\frac{k}{2}\log n$" form, which he proved in [15] to hold in general (i.e. not just for the models assumed here.)] Combining this with the expression for $f(\mathbf{y}|\hat{\theta}^{(\xi)}(\mathbf{y}))$ gives:

$$\hat{f}(\mathbf{y}) \;=\; \left\{\frac{\left(\frac{2\pi e}{n}\right)^{\frac{n}{2}} m^2\Gamma(\frac{m}{2})\Gamma((n-m)/2)}{4\left[\left(\frac{\mathsf{d}}{2\hat{\sigma}_0}\right)^m - 1\right]}\right\}\cdot[\hat{\sigma}(\mathbf{y})]^{-n} \tag{65}$$

The log of equation (65) is the codelength we seek. [More accurately, the code length is the log of $\hat{f}$ plus log $||\delta\mathbf{y}||$, but this is a constant for a given problem and doesn't change with model structure.]

$$L_{\mathrm{ML}}(\mathbf{y}) \;=\; -\log\hat{f}(\mathbf{y}) \;=\; n\log\hat{\sigma}(\mathbf{y}) \;+\; \log\left\{\frac{4\left(\frac{n}{2\pi e}\right)^{\frac{n}{2}}\left[\left(\frac{\mathsf{d}}{2\hat{\sigma}_0}\right)^m - 1\right]}{m^2\Gamma(\frac{m}{2})\Gamma((n-m)/2)}\right\} \tag{66}$$

As a final comment, it is interesting to note that if we view this as the type of "two-part" *incompleteness* removal described by equation (14) in Section 4.1, where $L(\hat{\theta}^{(\hat{\xi})},\hat{\xi})$ is the negative logarithm of the prior of equation (19), then that prior is uniform with respect to the parameters $\eta^{(m)}$.

5.2 Incompleteness Removal Using a Uniform Prior

The preceeding sections culminated in the derivation of equation (66), which is the code length for the data $\mathbf{y}$ based on a ML code. We now explore an alternate means of *incompleteness* removal. At this point we return to equation (14) in Section 4.1, which is:

$$L_c(\mathbf{y}) \;=\; L_c(\mathbf{y}|\hat{\theta}^{(\hat{\xi})},\hat{\xi}) \;+\; L(\hat{\theta}^{(\hat{\xi})},\hat{\xi}). \tag{67}$$

In the following section (4.2), we showed that if a certain sufficiency condition holds (which it does for our Gaussian linear-regression models) that the data-encoding term in this equation reduces to:

$$L_c(\mathbf{y}|\hat{\theta}^{(\hat{\xi})},\hat{\xi}) \;=\; \log\left\|\Omega(\hat{\theta}^{(\hat{\xi})},\hat{\xi})\right\| \;-\; \log||\delta\mathbf{y}|| \tag{68}$$

[Actually, we argued that $||\delta\mathbf{y}||$ could be ignored but we will keep it for now.] In this section we obtain an expression for this and for the parameter encoding cost $L(\hat{\theta}^{(\hat{\xi})},\hat{\xi})$ assuming a uniform encoding scheme (uniform prior) for the parameters.

The parameters must be encoded to finite precision (quantized) and we will represent these quantized parameters by $\tilde{\theta}^{(\hat{\xi})}$. This is an important point that will be key in our development. In the previous equation (68) the dimension of the manifold $\Omega(\hat{\theta}^{(\hat{\xi})},\hat{\xi})$ is equal to the number of points minus the number of parameters $(n-m-1)$ in our case. When everything (data plus parameters) is quantized, however, the corresponding region is n-dimensional and we will represent it by $\tilde{\Omega}(\tilde{\theta}^{(\hat{\xi})},\hat{\xi})$. Its content $||\tilde{\Omega}(\tilde{\theta}^{(\hat{\xi})},\hat{\xi})||$ divided by $||\delta\mathbf{y}||$ is equal to the number of data "states" (values of $\mathbf{y}$) whose ML parameter estimates $\hat{\theta}^{(\hat{\xi})}(\mathbf{y})$ quantize to $\tilde{\theta}^{(\hat{\xi})}$.

In what follows we will encode the parameters $(\eta^{(m)}, \rho)$, which are equivalent to the more usual $(\alpha^{(m)}, \hat{\sigma})$. We choose this representation because its associated coordinate transformations have convienient properties and because its direct geometrical interpretation facilitates geometrical reasoning and analogies.

5.2.1 A Simple Example: Three Data Points $(n = 3, m = 1)$

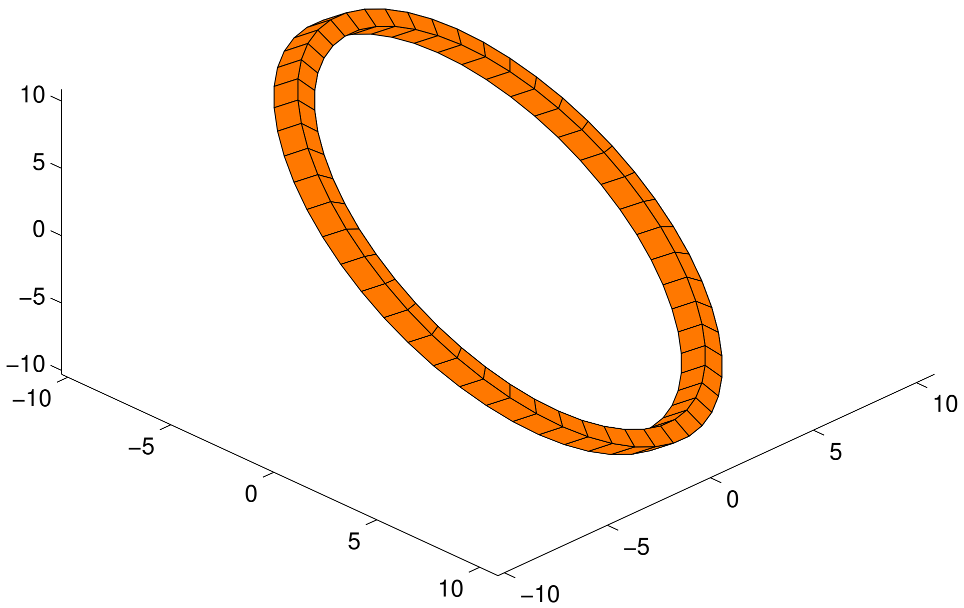


Figure 2: A visualization of the annular-cylindrical shell corresponding to quantized $\hat{\mu}$ and $\hat{\sigma}$ values in our three-dimensional example.

As we did for our ML-code-based formula we begin with a simple three-dimensional case like that depicted in Figure 1. Previously, we discussed the fact that in the continuous case the manifold of all data values corresponding to a given $(\hat{\mu}, \hat{\sigma})$ pair is a circle that is the intersection of the plane and the cylinder in Figure 1. For quantized $(\hat{\mu}, \hat{\sigma})$ the corresponding region of $\mathbf{y}$ values is the union of all such circles corresponding to all the $(\hat{\mu}, \hat{\sigma})$ values that quantize to the given quantized values. This region is depicted in Figure 2. The height of the shell is equal to the encoding precision $\delta\zeta = \delta\hat{\mu}$, while its thickness is the precision $\delta\rho$. The volume of this annular ring can be approximated by the perimeter of the associated circle times $\delta\zeta \cdot \delta\rho$. That is:

$$\left\|\tilde{\Omega}(\tilde{\theta}^{(\hat{\xi})}, \hat{\xi})\right\| = \int_{\tilde{\Omega}(\tilde{\theta}^{(\hat{\xi})}, \hat{\xi})} d\mathbf{y} = \delta\zeta \cdot \delta\rho \cdot \rho \cdot \int d\beta = (2\pi\rho) \cdot \delta\zeta \cdot \delta\rho, \tag{69}$$

5.2.2 Calculation of $\|\tilde{\Omega}(\tilde{\theta}^{(\hat{\xi})}, \hat{\xi})\|$ for the n-Dimensional Problem

Computing $\|\tilde{\Omega}(\tilde{\theta}^{(\hat{\xi})}, \hat{\xi})\|$ for our general n-dimensional case is similar to computing the integral in the denominator of our formula for $\hat{f}(\mathbf{y})$ and we will use exactly the same coordinate transformations, which yield:

$$||\tilde{\Omega}(\tilde{\theta}^{(\hat{\xi})}, \hat{\xi})|| \;=\; \int \rho^{n-m-1} d\rho \int C(\Theta) d\Theta \int d\eta^{(m)} \tag{70}$$

Because the ρ and $\eta^{(m)}$ integrals are only over the range of values corresponding to a single quantized parameter vector we will approximate them by what amounts to a first-order Taylor expansion, which gives:

$$||\tilde{\Omega}(\tilde{\theta}^{(\hat{\xi})}, \hat{\xi})|| \;=\; \delta\eta_1 \delta\eta_2 \ldots \delta\eta_m \delta\rho \rho^{n-m-1} \int C(\Theta) d\Theta. \tag{71}$$

Evaluating this gives a number of data states equal to:

$$\frac{||\tilde{\Omega}(\tilde{\theta}^{(\hat{\xi})}, \hat{\xi})||}{||\delta \mathbf{y}||} \;=\; \left[\frac{2\pi^{(n-m)/2}\rho^{n-m-1}}{\Gamma((n-m)/2)}\right] \cdot \frac{\delta\eta_1 \delta\eta_2 \ldots \delta\eta_m \delta\rho}{||\delta \mathbf{y}||} \tag{72}$$

5.2.3 Encoding the Quantized Parameters

We will encode our parameters $(\eta^{(m)}, \rho)$ with a uniform distribution. This means that the code length is equal to the log of the number of quantized parameter values. If we define a region of possible parameter values, $\Gamma(\hat{\xi})$ (i.e. $\theta^{(\hat{\xi})} \in \Gamma(\hat{\xi})$) [not to be confused with the "Gamma" function] then the number of such "states" is equal to the content of $\Gamma(\hat{\xi})$ divided by the content of one parameter quantization cell:

$$\frac{||\Gamma(\hat{\xi})||}{\delta\eta_1 \delta\eta_2 \ldots \delta\eta_m \delta\rho}. \tag{73}$$

The region $\Gamma(\hat{\xi})$ is really determined by Λ, the region (set) of possible data values $\mathbf{y}$. Following the assumption we used in developing our ML-code-based formula, we will assume that $\rho \in [\sqrt{n}\,\hat{\sigma}_0, \sqrt{n}\,(\mathsf{d}/2)]$, and that the region of possible $\eta^{(m)}$ values is the projection of the circumscribing sphere of the data cube onto the $\eta^{(m)}$ subspace $R\left(\Phi^{(m)}\right)$. The content of this $\eta^{(m)}$ region was computed in the previous section (see equation (62)). Using this, our parameter encoding length is given by:

$$L(\hat{\theta}^{(\hat{\xi})}, \hat{\xi}) \;=\; \log\left\{\frac{\sqrt{n}[(\mathsf{d}/2) - \hat{\sigma}_0] 2\pi^{\frac{m}{2}} n^{\frac{m}{2}} (\mathsf{d}/2)^m}{m\Gamma(\frac{m}{2})\delta\eta_1 \delta\eta_2 \ldots \delta\eta_m \delta\rho}\right\} \tag{74}$$

Our total encoding length is given by:

$$L_{\mathsf{c}} \;=\; \log\frac{||\tilde{\Omega}(\tilde{\theta}^{(\hat{\xi})}, \hat{\xi})||}{||\delta \mathbf{y}||} \;+\; \log\frac{||\Gamma(\hat{\xi})||}{(\delta\eta_1 \delta\eta_2 \ldots \delta\eta_m \delta\rho)} \tag{75}$$

In evaluating this expression further something significant becomes apparent. Namely, that the parameter encoding precisions from the two terms cancel [See equation (72).], leaving a result that is independent of this encoding precision. Performing this cancelation and dropping the $\log ||\delta \mathbf{y}||$ term as discussed previously, we obtain:

$$L_{\mathsf{c}} \;=\; \log ||\tilde{\Omega}(\tilde{\theta}^{(\hat{\xi})}, \hat{\xi})|| \;+\; \log ||\Gamma(\hat{\xi})||. \tag{76}$$

Substituting the expressions we obtained for these two quantities yields:

$$L_{\mathsf{c}}(\mathbf{y}) \;=\; \log\left\{\frac{2\pi^{(n-m)/2}[\sqrt{n}\hat{\sigma}(\mathbf{y})]^{n-m-1}}{\Gamma((n-m)/2)}\right\} \;+\; \log\left\{\frac{\sqrt{n}[(\mathsf{d}/2) - \hat{\sigma}_0] 2\pi^{\frac{m}{2}} n^{\frac{m}{2}} (\mathsf{d}/2)^m}{m\Gamma(\frac{m}{2})}\right\}, \tag{77}$$

which simplifies to:

$$L_c(\mathbf{y}) \;=\; \log\left\{\frac{4\pi^{n/2}n^{n/2}(\mathrm{d}/2)^m[(\mathrm{d}/2)-\hat{\sigma}_0]}{m\Gamma((n-m)/2)\Gamma(\frac{m}{2})}\right\} \;+\; (n-m-1)\log\hat{\sigma}(\mathbf{y}). \tag{78}$$

[The *Gamma* in these two formulas ((77) and (78)) represents the "Gamma" function (related to the *factorial* function and not the set of parameter values $\Gamma(\hat{\xi})$.]

5.2.4 Unlimited Data Values

Encoding parameters with a uniform prior is fine as long as $\Gamma(\xi)$ is finite. If it isn't, however, this strategy leads to infinite code lengths. Even if we assume that $\theta^{(\xi)}$ can take any value, our set of quantized values $\tilde{\theta}^{(\xi)}$ is countable which means that it can be associated with the positive integers. Thus, if we have a scheme for encoding the positive integers $\{1, 2, \ldots, \infty\}$, it can be used for encoding $\{\tilde{\theta}^{(\xi)}\}$. In the problem we are considering if the set of possible data values is unlimited (i.e. a quantized version of: $\mathbf{y} \in \Lambda = \mathbb{R}^n$), then the set $\Gamma(\xi)$ of possible parameter values is unlimited also. For this case we will follow Nohre and Rissanen before him by using Rissanen's universal prior for the positive integers, which will be described in the following section (Section 6). Here we encode two integers; one corresponding to $\eta^{(m)}$ and one to $\hat{\sigma}$ (actually ρ). The parameter coding length obtained is:

$$L(\tilde{\theta}^{(\xi)}|\xi) \;=\; L_u(\eta^{(m)}) \;+\; L_u(\rho/2). \tag{79}$$

where $L_u(\ldots)$ is defined in Section 6. This gives a total encoding length of:

$$L_c(\mathbf{y}) \;=\; \log\left\{\frac{2\pi^{(n-m)/2}[\sqrt{n}\hat{\sigma}(\mathbf{y})]^{n-m-1}}{\Gamma((n-m)/2)}\right\} \;+\; L_u(\eta^{(m)}) \;+\; L_u(\rho/2). \tag{80}$$

6 The Small-Sample Formula of Nohre

In [12, Chapter 2 "Coding Small Data Sets"] Nohre addresses the the same small-sample problem for the case of polynomial curve fitting. While his treatment is based explicitly on the QR decomposition[18, Section 4.4, p.195] it amounts to using the first of the two coordinate transformations we have used here and his analysis does in fact produce a *complete* code.

The coding scheme, which assumes an unlimited range of data values, encodes $\mathbf{y}$ by encoding $\eta^{(m)}$ and $\eta^{(m,n)}$ individually. It uses the same general encoding scheme for both. The resulting formula is:

$$L_{\mathrm{Nohre}}(\mathbf{y}|m) \;=\; L_u(\eta^{(m)}) + L_u(\eta^{(m,n)}), \tag{81}$$

where:

- $L_u(\mathbf{v}) \;\equiv\; L_0(V_S(\mathbf{v}))$, where $L_0(x)$ is Rissanen's universal prior for integers[13] (discussed below):

$$L_0(x) \;=\; \log^*(x) \;+\; C_0, \tag{82}$$

- $C_0 \;=\; 2.865\ldots$, and $V_S(\mathbf{v})$ is the volume of a sphere whose dimension is the number of components in the vector $\mathbf{v}$ and whose radius is $|\mathbf{v}|$.

- Equation (81) implicitly assumes that the quantized η are integer-valued (i.e. their quantization cells are unit hypercubes). A credible case can be made for this in the case of integer-valued $\mathbf{y}$. If this is not the case, $V_S(\mathbf{v})$ in these formulas must be divided by the volume of a η quantization cell.

The function $L_u(\ldots)$ was first used by Rissanen[13]. Its use by Nohre can be explained as follows. Associated with the partition of η into $(\eta^{(m)}, \eta^{(m,n)})$ are two discrete grids (corresponding to the quantized data/parameters) - one imbeded in $\mathbb{R}^{m+1}$ and one in $\mathbb{R}^{n-m}$. These grids each contain an infinite number of points. In each case we (Nohre) would like to construct a coding scheme that weights the points uniformly (i.e. a uniform prior). Unfortunately, because there are an infinite number, a uniform encoding would result in infinite code lengths. There exists, however, an encoding of the positive integers (due to Elias[3]) that is in some sense as close as one can get to a uniform encoding while still providing a finite-length code for any integer. [For a fascinating treatment of this and related schemes see [9].]. Using this scheme to construct a normalized prior of the form $\pi(x) = 2^{-L(x)}$[13] yields the code length of equation (82).

To utilize this scheme in our case, we must put our η grid points in a one-to-one correspondence with the positive integers. In the integer-data case that Nohre assumes the η points are on a square lattice with unit spacing between grid points. Thus the number of points in any region (of $\mathbb{R}^{m+1}$) is given approximately by its content. We associate these with positive integers as follows. We start by numbering the point at the origin (e.g. $\eta^{(m)} = 0$) as 1 and then "spiral" outward, sequentially numbering all points in a given "shell" (constant distance (e.g. $\|\eta^{(m)}\|$) from the origin), assigning the next available integer to the first point in the next farther-out shell, and so on. Using this scheme, a point's integer index can be approximated by the volume of the sphere (centered at the origin) on whose surface the point lies.

7 Experimental Evaluation

To evaluate the effectiveness of the criteria investigated here, simulation experiments were run. In doing this two questions arise:

1. how to evaluate the effectiveness of the techniques

2. what experiments to run

7.1 Effectiveness Measures

We will be concerned here with measures that might be termed "accuracy" and not with computational considerations, though, there turns out to be little difference in computational cost among the techniques. The relevant effectiveness measure to use will depend to a great extent on the intended use of the fitted polynomials. If, for example, the intended use were to do data compression in the assumed manner, code length would be the appropriate measure. While we report code lengths, we assume that the most important issue in most applications is the question of how well the estimated polynomial matches the "true" polynomial. To assess and characterize this we compute the following error measure, which we refer to as "prediction error":

$$\epsilon = \sqrt{\frac{1}{a-b}\int_b^a [p_f(x) - p_t(x)]^2 dx}, \tag{83}$$

where $[a, b]$ is the range of x values covered by the data, $p_f(x)$ is the fitted polynomial and $p_t(x)$ is the true polynomial. Computing this measure, automatically weights the errors by their severity. It should be emphasized that an exact expression is computed for ϵ and not just the sum of the errors for the fitted data.

This *prediction-error* measure ϵ weights the model-order selection errors by their severity. Occasionally, the associated penality can be quite severe, for example, in cases where there is a small number of points and a high-degree polynomial is chosen, the polynomial may vary widely from the true function between the points. Partially for this reason, we also include histograms of the polynomial degrees selected by the various techniques studied for the various techniques/formulas considered.

7.2 Simulation Experiments

Simulation experiments were run to compare the performance of the following five code-length formulas when used as objective functions for estimating model order in linear regression.

1. The "classic" MDL form which has a parameter code length of $\frac{k}{2}\log n$ and no *incompleteness* removal.

2. Nohre's method.

3. Our *uniform-prior* form for limited data values.

4. Our *uniform-prior* form for unlimited data values $(-\infty, +\infty)$ which uses Nohre's parameter encoding cost.

5. The formula we derived based on maximum-likelihood encoding.

The linear-regression problem considered is that of polynomial curve fitting. Experiments were run involving a vector $\mathbf{x}$ of values of a scalar independent variable x and a vector $\mathbf{y}$ of a scalar dependent variable y. Five polynomials were used covering polynomial degrees 0,1,2,3 and 4. Four of these polynomials are graphed in Figure 3 over the x ranges used in the experiments. The fifth is the 0^{th}-degree polynomial: $y = 127$. The actual polynomial coefficients are tabulated along with detailed summaries of the experimental results in Appendix D. For all but the 2^{nd}-degree case the data was constrained such that $x \in [0, 511]$ and $y \in [0, 255]$. These ranges were chosen because they are common in the digital image-analysis applications that motivated this work. The ranges for the 2^{nd}-degree case were chosen to match those used in [12] as was the polynomial itself.

The "atomic" experiments making up this study consisted of taking one of the polynomials, adding uncorrelated zero-mean Gaussian noise of a given standard deviation σ, and determining the optimal model - $(\hat{\theta}^{(\hat{\xi})}, \hat{\xi})$ (i.e. the polynomial degree and coefficients). This was carried out by fitting polynomials of all degrees, zero through six (or two less than the number of points, which

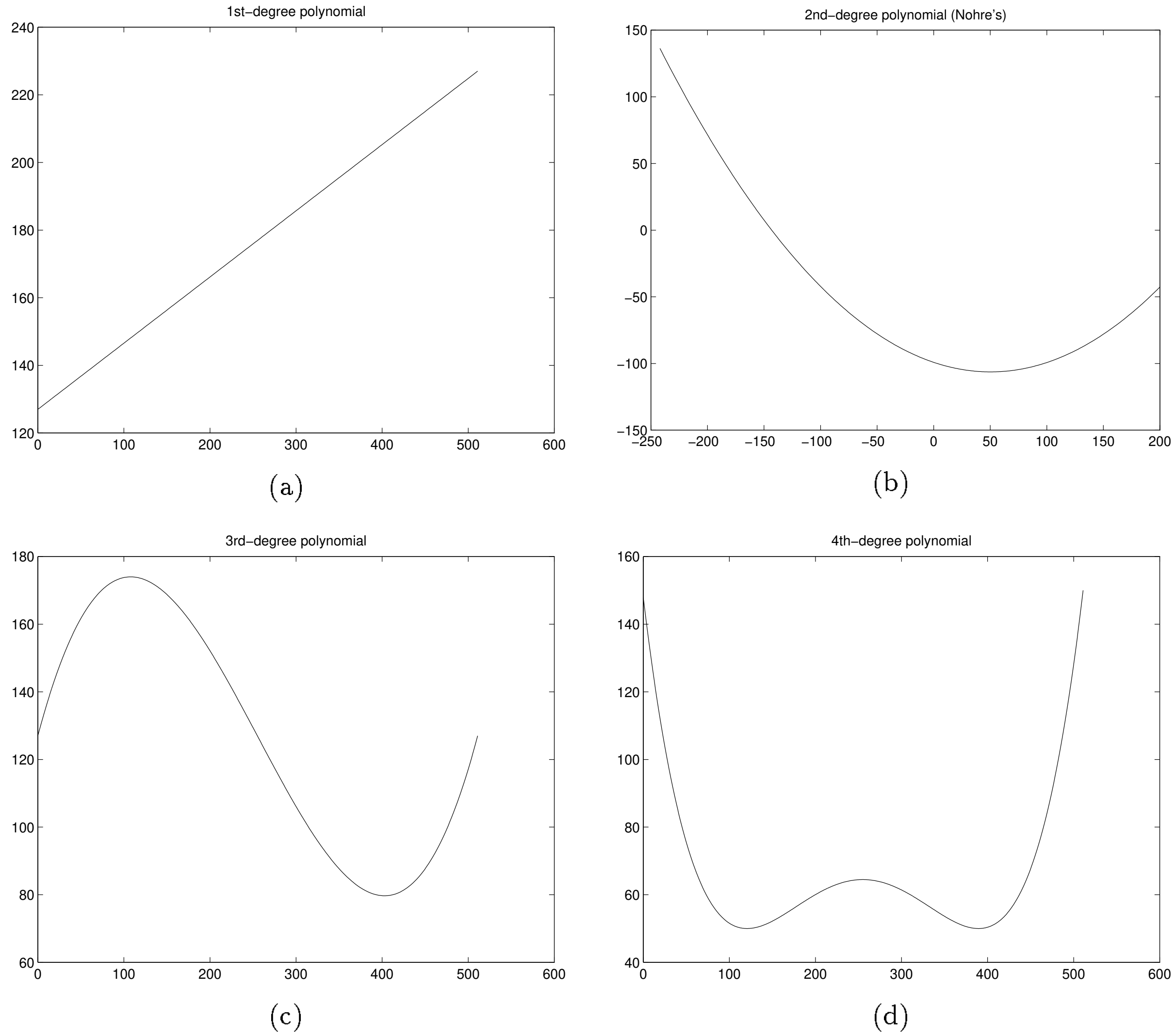


Figure 3: This figure contains graphs of four of the five polynomials used in the simulation experiments presented here. The fifth function was the simple constant: $y = 127$. These are: (a) 1^{st} degree; (b) 2^{nd} degree; (c) 3^{rd} degree; (d) 4^{th} degree.

ever was less) and evaluating the various code-length functions for each fit. The degree/fit with the smallest code length was determined for each of the five code-length functions discussed here.

A matrix of experiments was performed using each of the five polynomials of Figure 3 over the data ranges shown. The experiments correspond to all possible pairs of one of five σ values $\{3, 5, 10, 20, 40\}$, paired with one of six numbers of points (n_p): $\{7, 10, 20, 40, 60, 80\}$. In addition a second matrix was run for only the 4^{th}-degree polynomial. This matrix uses the same set of σ values but only the following n_p values: $\{5, 6, 7, 8, 9, 10\}$. [In terms of the code-length formulas n and n_p are identical. A different symbol, n_p was used because this is an experimental parameter.] This was performed to study the small-sample behavior in more detail for one of the polynomials. At each (polynomial, σ, n_p) combination, 100 vectors of n_p random integer x values (with no duplication) were generated. For each of these x values the value of the polynomial was calculated and then a random number from the described distribution was added to this and the result was rounded to the nearest integer to obtain the corresponding y value. Values of y falling outside of the $[0, 511]$

range were "clipped" to either 0 or 511, whichever was closer. The various parameters and ranges were chosen to ensure that this would be a relatively rare occurrence. The values of these known limits on y were used in the two code-length formulas that assume limited values - the uniform-prior/limited-values form and the maximum-likelihood form. Also, as mentioned previously, the limit $\hat{\sigma}_0$ on $\hat{\sigma}(\mathbf{y})$ was set to the data quantization, which in these experiments was equal to 1.

7.3 Summary of Results

As mentioned previously, a tabulated detailed summary of the experimental results can be found in Appendix D. Three of those tables are included here as representative of the overall performance of the techniques. These correspond to the 2^{nd}-degree polynomial (Nohre's). Table 1 contains prediction-errors for all the (σ, n_p) combinations, Table 2 and Figure 4 present the prediction-error results for all techniques in the 7-point case, where the difference is most significant, and Table 3 shows histograms of the polynomial degrees selected by the various techniques in the 7-point case. [These tables are repeated in the Appendix.]

We present here a further summary of these results. The following comments are based on the tabulated prediction-error, order-selection and code-length results. For all techniques we see the expected improvement in order selections with increasing n_p and the expected degradation with increasing σ. We attribute the occasional exception to this trend to statistical variation. For large n_p (e.g. $n_p = 80$) and small σ the techniques all performed well with the two uniform-prior based schemes and Nohre's scheme giving identical results. The classic $(k/2)\log n$ and the maximum-likelihood formulas did slightly worse there, with the edge going to the maximum-likelihood scheme.

For small n_p, on the other hand, all the *complete* schemes did significantly better than the classical $(k/2)\log n$ formula and the uniform-prior-based approaches (including Nohre's) did significantly better than the maximum-likelihood approach. Finally, at $n_p = 7$, The limited-values formula did better than unlimited, which did better than Nohre's. A slight exception is observed for the 3^{rd} and 4^{th}-degree polynomials at $n_p = 7$. There, Nohre's formula and the uniform-prior-unlimited formula show a stronger dependence on σ than the limited-values formula. In that case $(n_p = 7)$ the former techniques do better for small σ and worse for large σ. A more detailed look at how the five techniques perform for small n is given by the results in Appendix D.5.2. In interpreting these one should keep in mind the limit on maximum order applied; that is, the highest polynomial degree considered is $\min((n_p - 2), 6)$. An examination of the associated order-selection results in Tables 51 through 55 shows that all the techniques except the limited-values approach chose the largest degree available on some non-insignificant fraction of the trials. For this reason, if these techniques had been allowed to choose higher orders, a bigger difference in prediction error would have been observed between the limited-values formula and the others.

The tabulated encoding lengths confirm the expected behavior. The classical $(k/2)\log n$ form must be excluded from these comparisons, however. It gives lower code-length numbers than the other approaches and this is to be expected because it is an asymptotic form that neglects terms that become important for small n_p. Of the other techniques, the limited-values formula gives consistently lower code lengths than the unlimited-values formula, which gives lower values than Nohre's formula. The maximum-likelihood code lengths show a stronger dependence on σ than the others, giving smaller code lengths than the others for small σ and larger values than the others in

Table 1: Prediction error results for the five code-length formulas investigated here when applied to the 2^{nd}-degree polynomial shown in Figure 3 (b).

$\sigma \downarrow \ n_p \rightarrow$	7	10	20	40	60	80
$\frac{k}{2}\log n$ (Rissanen)						
3	26.0353	10.1129	1.3351	0.8715	0.6832	0.5487
5	24.1208	9.6496	2.4987	1.3129	1.1333	0.8949
10	65.4654	18.8983	3.7171	2.8059	2.2222	1.8413
20	200.5477	22.0454	8.3216	5.4536	4.4471	3.9510
40	233.2479	81.4801	16.6584	11.2928	8.6327	7.0955
Nohre						
3	1.7420	1.5173	1.0961	0.7783	0.6223	0.5078
5	3.7917	2.2692	1.6917	1.1808	1.0551	0.8740
10	7.7640	5.1681	3.1244	2.6373	2.1147	1.7697
20	128.5057	10.5525	6.9196	5.1657	4.2107	3.7770
40	160.9505	43.6533	14.4804	10.7400	8.4171	6.9656
Uniform Prior (limited)						
3	1.6738	1.5210	1.0961	0.7783	0.6223	0.5078
5	2.9749	2.2489	1.6917	1.1808	1.0420	0.8740
10	6.2162	5.1281	3.1054	2.6373	2.1147	1.7697
20	12.9507	9.4925	6.7314	5.1657	4.1299	3.7071
40	28.0298	18.9391	13.7594	10.5135	8.2701	6.9250
Uniform Prior (unlimited)						
3	1.7348	1.5173	1.0961	0.7783	0.6223	0.5078
5	3.1256	2.2692	1.6917	1.1808	1.0551	0.8740
10	7.3170	5.1681	3.1244	2.6373	2.1147	1.7697
20	112.3571	10.5525	6.9196	5.1657	4.2107	3.7770
40	87.4595	43.8551	14.4804	10.7400	8.4171	6.9656
Maximum-Likelihood						
3	9.3982	6.3082	1.1411	0.7844	0.6313	0.5215
5	18.3507	4.1464	1.7493	1.1808	1.0728	0.8814
10	43.9266	12.9325	3.2709	2.6373	2.1147	1.7894
20	138.5692	13.1315	7.0010	5.1657	4.2107	3.7770
40	210.3525	45.7414	14.0504	10.5135	8.2065	6.9250

almost all cases for large σ.

In an attempt to summarize these results in a single sentence, we would say that all the *complete* techniques gave results clearly superior to those obtained using the classic form, with the ranking being: limited-values, unlimited-values, followed by the other two (Nohre and ML). The superiority of the limited-values formula was the greatest at very small sample sizes.

Table 2: Prediction error results for the five code-length formulas investigated here when applied to the 2^{nd}-degree polynomial shown in Figure 3 (b). These results are for 7-data-points case only. These results are also presented in graphical form in Figure 4.

$\sigma \downarrow$ *method* $\rightarrow$	Rissanen ($\frac{k}{2}\log n$)	Nohre	UP-UL	UP-LIM	ML
3	26.0353	1.7420	1.7348	1.6738	9.3982
5	24.1208	3.7917	3.1256	2.9749	18.3507
10	65.4654	7.7640	7.3170	6.2162	43.9266
20	200.5477	128.5057	112.3571	12.9507	138.5692
40	233.2479	160.9505	87.4595	28.0298	210.3525

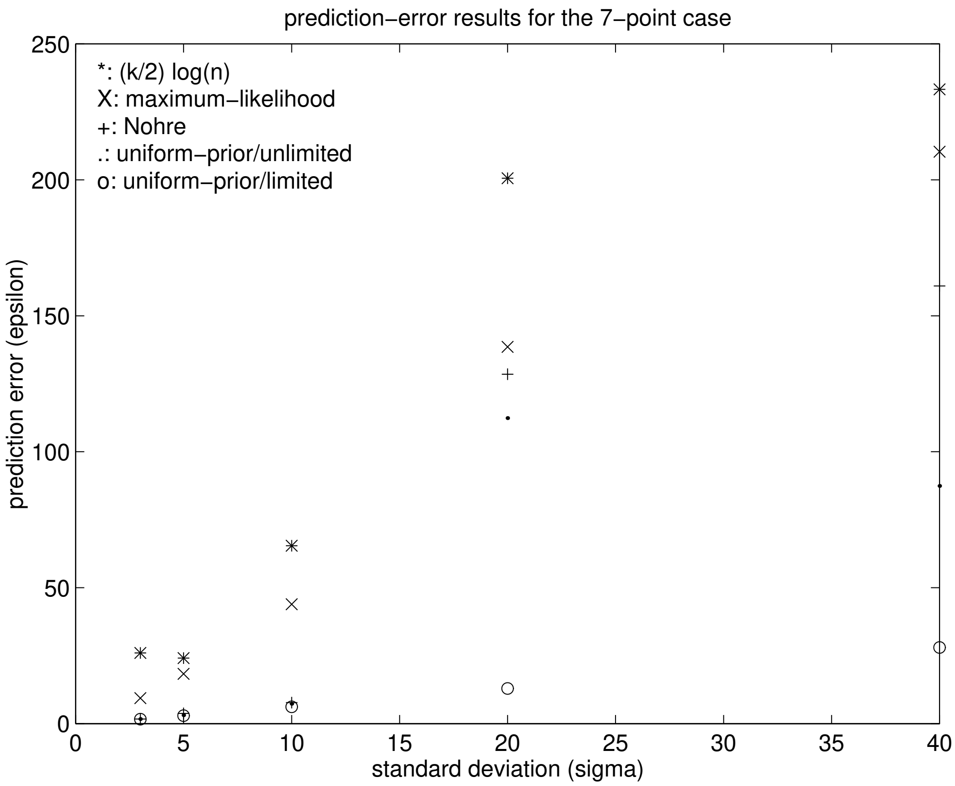


Figure 4: Prediction error results for the five code-length formulas investigated here when applied to the 2^{nd}-degree polynomial shown in Figure 3 (b). These results are for 7-data-points case only. The legend in the upper-left corner of the figure identifies the correspondence between plotting symbols and code-length formulas. These results are tabulated in Table 2.

8 Summary, Analysis and Conclusions

8.1 Order Estimation and Prediction Error

By correcting for coding *incompleteness* we have obtained non-asymptotic code-length formulas for the problem of linear regression with order determination. In our polynomial-curve-fitting simulation experiments they performed significantly better at order estimation for small sample sizes than the classic asymptotic $(k/2)\log n$ formula of Rissanen[13], as expected. Among the

Table 3: Polynomial degrees selected by various methods for the 7-point case and the 2^{nd}-degree polynomial.

degree ↘	Rissanen ($\frac{k}{2}\log n$)							Nohre						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	21	6	12	61	0	0	0	96	3	1	0	0
5	0	0	14	10	9	67	0	0	0	95	1	3	1	0
10	0	0	15	7	8	70	0	1	0	90	6	2	1	0
20	0	0	20	10	14	56	0	2	0	72	8	7	11	0
40	0	0	15	4	20	61	0	12	5	43	8	8	24	0

degree ↘	Uniform Prior (unlim.)							Uniform Prior (limited)							Maximum-Likelihood						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	97	3	0	0	0	0	0	100	0	0	0	0	0	0	51	7	6	36	0
5	0	0	95	5	0	0	0	0	0	100	0	0	0	0	0	0	48	5	4	43	0
10	1	0	92	6	1	0	0	0	0	99	1	0	0	0	0	0	50	6	6	38	0
20	3	0	76	8	6	7	0	1	0	98	1	0	0	0	0	0	53	4	12	31	0
40	17	5	52	9	8	9	0	4	14	80	2	0	0	0	3	6	30	5	12	44	0

formulas investigated, the one that worked best in our experiments was the one that corresponds to encoding the model parameters based on a uniform prior and assuming a known limited range of data values (the "uniform-prior/limited-values" case). The limits of this range were arguments of the code-length formula. The advantage of using this formula was most pronounced for the smallest sample sizes as we expected. The second best results were obtained with the "uniform-prior/unlimited-values" model, which makes no assumptions about the range of data values. The following is a list of equations that specify the formulas to use with the various techniques.

1. classic asymptotic $(k/2)\log n$: equation (12)
2. maximum-likelihood code: equation (66)
3. uniform-prior/limited-values: equation (78)
4. uniform-prior/unlimited-values: equation (80)
5. Nohre's formula: equation (81)

It should be pointed out that we have no theoretical results that guarantee the general superiority of any of these formulas over any of the others in estimating model orders. The philosophical foundation of MDL includes the belief that encoding schemes giving the smallest code lengths (assuming the associated code-length estimates are accurate, of course) will be the best predictors. This statement (ours) of the belief, however, begs the question: "Under what conditions?". All of the four *complete* techniques that we investigated can be considered to assume certain models for the data generating process. Taking a Bayesian view, the two "uniform-prior" schemes and the maximum-likelihood approach assume polynomials plus uncorrelated Gaussian noise with the parameters (σ, α) being generated by a "source" whose distribution we identify with the Bayesian notion of a "prior". In each of these three cases this prior is different. Our experiments have obviously covered a quite limited range of possible polynomial degrees and (σ, α) values. One might speculate that, in a large number of experiments in which the model parameters were generated from such a prior distribution, the encoding technique whose associated prior best matches this generating prior would give the best results.

Taking this sort of view, reveals Nohre's formula to be unique among those we investigated in the following sense. While he claims to assume Gaussian models, his scheme doesn't, in fact, make use of this assumption. Both the parameters and the data (residuals) are encoded using the function $L_u(\ldots)$, which embodies Rissanen's universal prior for integers. This leads us to a surprising conclusion. In our view, the function $L_u(\ldots)$ (or we should say the underlying coding scheme) is being used because we would like to use a uniform-length encoding, but this would lead to infinite code lengths when the range of possible data/parameter values is unlimited. In such a case the closest we can come (in some sense) to a uniform encoding while keeping the code lengths finite is to use the scheme associated with $L_u(\ldots)$. Thus, when we do have limited data values, we would simply use a uniform prior where Nohre has used $L_u(\ldots)$. Going through that exercise leads to a code length formula that is independent of the data values $\mathbf{x}$ and $\mathbf{y}$, depending only on m, n and d and therefore having no power to estimate the optimal model order m. It appears to be only the weak variation of $L_u(\mathbf{v})$ with $\|\mathbf{v}\|$ that provides this discrimination ability.

8.2 Code Lengths and Their Implications for Piece-Wise Fitting

As explained in the experimental results section (Section 7) the code lengths obtained experimentally agree with what would have been expected on theoretical grounds. The actual code lengths obtained have no direct significance in the problem treated here, however. It is only the ability of the associated formulas to select model orders that matters. Thus, adding a large constant to the code length, for example, would create no problems. These code lengths are important, however, in certain other problems. For example, in piecewise curve fitting (e.g. spline fitting) where the number of curve segments is part of the model structure ξ, such constants can, depending on their size, have a strong influence on the cost of adding another curve segment.

8.3 Possible Future Work

We envision the following investigations as potential future work for ourselves and others.

1. Expand theoretical and experimental analyses to better understand the behavior of the various formulas investigated here.

2. Attempt to obtain more accurate forms of the code-length formulas assuming limited values by improving the limits of integration (assumed region Λ).

3. Extend these results to vector-valued y.

4. Extend to piecewise-polynomial (e.g. splines) curve fitting.

5. Apply these results to surface fitting and other linear-least-squares problems.

6. Apply these results to the problem of image segmentation as in [6, 7].

Acknowledgment

Thanks to especially to Jorma Rissanen for commenting on multiple drafts of this report and making many helpful suggestions. Thanks also to Eric Pauwels for many helpful discussions and suggestions throughout the course of this work. Thanks also to Ron Fagin and Jianching Mao for reading and commenting on a draft of this report. Finally, last but not least, thanks to K.U. Leuven for hosting me and providing financial and other support on the one-year sabbatical, during which I carried out the bulk of the work described here. Thanks also to the faculty who invited, sponsored and provided other support to me. Formost among these I would like to single out Professors Andre Oosterlinck, Partrick Wambacq and Luc Van Gool.

References

[1] James O. Berger. *Statistical Decision Theory and Bayesian Analysis.* Springer-Verlag, New York; Berlin, 1985. ISBN: 0-387-96098-8.

[2] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* Wiley, New York, 1991.

[3] P. Elias. Universal codeword sets and representations of the integers. *IEEE Transactions on Information Theory*, 2, 1975.

[4] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics.* Addison-Wesley, New York, 1989. ISBN 0-201-14236-8.

[5] H. L. Gray and W. R. Schucany. *The Generalized Jackknife Statistic.* Marcel Dekker, New York, 1972.

[6] T. Kanungo, B. Dom, W. Niblack, and D. Steele. A fast algorithm for MDL-based multi-band image segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 609–616, Seattle, Washington, June 1994. IEEE Computer Society.

[7] T. Kanungo, B. Dom, W. Niblack, and D. Steele. MDL-based multi-band image segmentation using a fast region merging scheme. Technical Report 9960, IBM Research Division, May 1995.

[8] Maurice G. Kendall. *A Course in the Geometry of N Dimensions.* Griffin's Statistical Monographs and Courses. Charles Griffin, London, 1961.

[9] Donald E. Knuth. Supernatural numbers. In David A. Klarner, editor, *The Mathematical Gardner*, pages 310–325. Wadsworth International, Belmont, California, 1981.

[10] Y. G. Leclerc. Constructing simple stable descriptions for image partitioning. *International Journal of Computer Vision*, 3:73–102, 1989.

[11] F. Mosteller and J. W. Tukey. *Data Analysis and Statistics.* Addison-Wesley, 1968.

[12] Ragnar Nohre. *Topics in Descriptive Complexity.* PhD thesis, Technical University of Linkoping, 1993.

[13] J. Rissanen. A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, 2(11):211–222, 1983.

[14] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*, volume 15. World Scientific Series in Computer Science, 1989.

[15] J. Rissanen. Fisher information and stochastic complexity. *IEEE Trans Inf Th*, 42(1):40–47, January 1995.

[16] C. E. Shannon. A mathematical theory of communication. *Bell Syst Tech J.*, (3):379–423, 1948.

[17] James A. Storer. *Data Compression: Methods and Theory.* Computer Science Press, Rockville, MD, 1988. ISBN: 0-7167-8156-5.

[18] G. Strang. *Introduction to Linear Algebra.* Wellesley-Cambridge Press, Wellesley, MA, 1993. ISBN: 0-9614088-5-5.

APPENDICES

A Completeness in Coding

This appendix addresses the issue of *completeness* in coding theory as it relates to the problem treated by this paper. A code that is *incomplete* is also *redundant*, but the converse doesn't necessarily hold. The property of *redundancy* is defined for a given source/code combination in terms of the *entropy* H of a source, which is the shortest average per-symbol code length that is theoretically achievable. If we let L represent the actual average code length obtained for the source with a given encoding scheme, then the redundancy is defined as $L - H$. If $L > H$ for a given source/code combination, then the code is is *redundant* and, in a sense, contains a certain number of unnecessary bits. In many cases of redundant codes it is impossible to identify specific bits as redundant, however. In the case of an *incomplete* code, on the other hand, this identification is possible.

A binary tree is said to be *complete* if every node is either a leaf or an internal (decision) note with two children. A binary *prefix* code is decodable by a binary tree and if the associated decoder tree is *complete* then the Kraft inequality[2, Section 5.2 p.82] holds with equality:

$$\sum_{\text{all symbols i}} 2^{-L(i)} = 1, \tag{84}$$

where $L(i)$ is the length (in bits) of the code for symbol i, and the code is said to be *complete*.

In the main text of this paper we identified a certain type of *incompleteness* encountered in two-part coding. We pointed out that the associated encoding scheme included codes that could never be encountered in practice. It included multiple codes for every data vector $\mathbf{y}$, but only one code would ever be chosen - the shortest one. Thus, if we prune all branches of the tree that can never be traversed (corresponding to these unused code strings) we will have the actual decoder tree and it will clearly be incomplete and it is thus in this sense that the corresponding two-part code is incomplete.

B Asymptotic Analysis of $\log \int_\Lambda f(\mathbf{y}|\hat{\theta}^{(\xi)}(\mathbf{y}))d\mathbf{y}$

Here we examine what happens to the logarithm of the denominator of equation (45) for our ML solution. (equation (64)) as $n \to \infty$. We repeat this equation here:

$$\int_\Lambda f(\mathbf{y}|\hat{\theta}^{(\xi)}(\mathbf{y}))d\mathbf{y} = \left[\frac{n^{\frac{n}{2}}}{(2e)^{\frac{n}{2}}\Gamma((n-m)/2)}\right]\left\{\frac{4}{2^m m^2 \Gamma\left(\frac{m}{2}\right)}\left[\left(\frac{\mathsf{d}}{\hat{\sigma}_0}\right)^m - 1\right]\right\} \tag{85}$$

Because we are only concerned with asymptotics ($n \to \infty$ for fixed m), we need only consider the n-dependent part of this expression (i.e. the left half in [...]). Taking the log of this term gives:

$$\frac{n}{2}\log\frac{n}{2} - \frac{n}{2}\log 2 - \frac{n}{2}\log e - \log\Gamma\left(\frac{n-m}{2}\right). \tag{86}$$

We will now take the asymptotic form of this, starting with the $\log\Gamma$ term, which we will expand using Stirling's approximation [See, for example, [4, p.467]] as $(n \to \infty)$:

$$\log\Gamma(x+1) \sim x\log x - x\log e + \frac{1}{2}\log x \tag{87}$$

$$\log\Gamma\left(\frac{n-m}{2}\right) \sim \frac{n-m-2}{2}\log\frac{n-m-2}{2} - \frac{n-m-2}{2}\log e + \frac{1}{2}\log\frac{n-m-2}{2} \tag{88}$$

Asymptotically, we can replace $\log(n-m-2)$ with $\log n$. Doing this, substituting the resulting expression for $\log\Gamma\left(\frac{n-m}{2}\right)$ in equation (86) and simplifying gives as $(n \to \infty)$:

$$\log\int_\Lambda f(\mathbf{y}|\hat{\theta}^{(\xi)}(\mathbf{y}))d\mathbf{y} \sim \frac{m+1}{2}\log n \tag{89}$$

which agrees with Rissanen's classic asymptotic form. Here, the number of adjustable parameters is $(m+1)$, corresponding to σ and the m polynomial coefficients. This agreement is reassuring because Rissanen proved in [15] that it should hold in general (i.e. not just for the models assumed here).

C Surfaces of Constant Variance in $\mathbb{R}^n$

We are interested in characterizing surfaces where $\hat{\sigma}$ is constant. These are surfaces in $\mathbb{R}^n$ corresponding to $\mathbf{y} \in \mathbb{R}^n$.

For a Gaussian distribution we can write the following.

$$\hat{\mu} = \tfrac{1}{n}\textstyle\sum_{i=1}^n x_i \tag{90}$$

$$\hat{\sigma} = \sqrt{\tfrac{1}{n}\textstyle\sum_{i=1}^n (x_i-\hat{\mu})^2} \tag{91}$$

$$\hat{\sigma} = \tfrac{1}{n}\sqrt{n\textstyle\sum_{i=1}^n x_i^2 - \left(\sum_{j=1}^n x_j\right)^2} \tag{92}$$

$$\hat{\sigma} = \sqrt{\textstyle\sum_{i=1}^{n-1}\sum_{j=i+1}^n (x_i-x_j)^2} \tag{93}$$

For a fixed, known μ [different from constant $\hat{\mu}(\mathbf{y})$] surfaces of constant $\hat{\sigma}(\mathbf{y})$ are hyperspheres centered on the point $x_\mu^n : x_1 = x_2 = \ldots = x_n = \mu$. Surfaces of constant $\hat{\mu}(\mathbf{y})$ (say $\hat{\mu}(\mathbf{y}) = \mu$, on the other hand are hyperplanes that are normal to the line $\{x_1 = x_2 = \ldots = x_n\}$ and include the point x_μ^n. Thus for a fixed $\hat{\mu}(\mathbf{y}) = \mu$, the loci of constant variance (i.e. so both $\hat{\mu}$ and $\hat{\sigma}$ are fixed) are $(n-1)$ dimensional hyperspheres (which are $(n-2)$-dimensional manifolds imbeded in $\mathbb{R}^n$) whose center is x_μ^n and which lie in the hyperplane $\hat{\mu}(\mathbf{y}) = \mu$. A surface of constant $\hat{\sigma}(\mathbf{y})$ is the surface traced out by these $(n-1)$-dimensional hyperspheres as we vary $\hat{\mu}$. These surfaces are clearly hyper-circular cylinders whose axis is the line $\{x_1 = x_2 = \ldots = x_n\}$ and whose radius is $\sqrt{n}$ times the fixed value of $\hat{\sigma}$.

D Experimental Results - Complete Tabulation

This section contains a tabulation of all the results of our simulation experiments. [There is nothing special about the 4-decimal-place precision of these results. That was simply what MATLAB output. The fact that so many of the results are identical might cause some concern. The reason for this is that, for a given model order, all the techniques use exactly the same least-squares technique to estimate the parameters.]

D.1 Constant "polynomial"

This set of experiments was performed on synthetic data generated using a 0^{th}-degree polynomial (i.e. a constant). The following is a list of the associated experimental parameters:

- Polynomial Coefficients:

  $C_0 = 127$

- (xmin, xmax, ymin, ymax) = [0, 511, 0, 255]
- number of trials at each (σ, n_p) pair = 100
- σ values = 3, 5, 10, 20, 40
- numbers of points (n_p) = 7, 10, 20, 40, 60, 80

Table 4: Prediction error results for the five code-length formulas investigated here.

$\sigma \downarrow \;\; n_p \rightarrow$	7	10	20	40	60	80
$\frac{k}{2}\log n$ (Rissanen)						
3	15.0286	11.3560	0.7940	0.5111	0.3464	0.3142
5	33.4121	9.4970	1.4154	0.7785	0.5807	0.4707
10	119.2872	17.1387	2.7844	1.2813	1.2872	1.0180
20	138.1347	18.4283	4.9501	3.0891	2.2744	2.0666
40	239.1173	78.2865	10.3502	6.2000	5.2641	4.3338
Nohre						
3	0.8958	0.6950	0.5550	0.4325	0.3262	0.2754
5	1.6195	1.2080	0.8995	0.6927	0.5002	0.4211
10	3.0109	2.9476	1.8695	1.1784	1.0887	0.8652
20	7.5651	5.2086	3.6200	2.7527	1.9287	1.8800
40	32.5131	10.7277	6.7240	5.2207	4.7653	3.8785
Uniform Prior (limited)						
3	0.8886	0.6950	0.5550	0.4325	0.3262	0.2754
5	1.6014	1.1770	0.8995	0.6927	0.5002	0.4211
10	3.0109	2.9476	1.8695	1.2033	1.0887	0.8652
20	7.2207	5.3639	3.6200	2.7527	1.9869	1.9243
40	14.0524	10.9881	7.1632	5.6927	4.9727	4.1362
Uniform Prior (unlimited)						
3	0.8958	0.6950	0.5550	0.4325	0.3262	0.2754
5	1.6195	1.2080	0.8995	0.6927	0.5002	0.4211
10	3.0109	2.9476	1.8695	1.1784	1.0887	0.8652
20	7.2207	5.2086	3.6200	2.7527	1.9287	1.8800
40	17.7251	10.7277	6.7240	5.2207	4.7653	3.8785
Maximum-Likelihood						
3	8.2361	9.7841	0.6072	0.4379	0.3262	0.3010
5	19.8609	3.3282	1.0350	0.7293	0.5242	0.4427
10	39.9395	5.3428	1.9170	1.2033	1.2052	0.9475
20	68.5997	6.2378	3.8745	2.8187	2.1312	1.9736
40	170.6604	24.4103	7.1632	5.6927	4.9727	4.1362

Table 5: Total encoding length (in bits; rounded to nearest bit) obtained using the encoding schemes investigated here.

$\sigma\downarrow\ n_p\rightarrow$	7	10	20	40	60	80
			$\frac{k}{2}\log n$ (Rissanen)			
3	18.8322	33.3641	69.8501	140.4936	209.9940	278.4834
5	22.2436	40.8811	85.1696	169.8458	253.2301	336.0913
10	28.5115	49.9931	106.0467	209.3584	313.8025	416.1632
20	36.5314	60.7538	125.3346	249.5331	373.8861	495.5400
40	43.5182	68.6703	145.9748	289.8372	433.4634	575.4650
			Nohre			
3	47.0766	59.8095	96.7567	172.1491	246.3996	319.5189
5	51.9649	66.9459	111.7696	201.1703	289.2514	376.7324
10	58.8626	75.7710	132.0558	240.1167	349.3089	456.2373
20	64.9155	85.1725	150.6010	279.5984	408.6927	534.8988
40	71.6949	94.2161	170.5767	319.2612	467.4952	614.1657
			Uniform Prior (limited)			
3	36.4700	48.3580	84.4858	159.1116	233.0036	305.9364
5	40.0786	54.2677	98.2915	187.0160	274.7900	362.0789
10	45.3136	61.6580	117.0609	224.5414	333.4121	440.1843
20	50.0008	69.6156	134.2960	262.6972	391.4603	517.5375
40	55.3644	77.3890	152.9109	301.0202	448.9991	595.4775
			Uniform Prior (unlimited)			
3	44.1808	56.6844	93.3315	168.6208	242.8836	316.0108
5	49.1716	63.9250	108.5104	197.8386	285.8877	373.3499
10	56.2850	72.9947	129.0611	236.9647	346.0946	452.9821
20	62.5587	82.6114	147.7635	276.5740	405.5854	531.7376
40	69.5627	91.7949	167.8607	316.3326	464.4697	611.0789
			Maximum-Likelihood			
3	16.7788	28.8663	52.2331	102.5298	151.3337	198.3982
5	24.5951	43.8180	82.9057	161.2109	237.8262	313.5445
10	37.2170	62.7316	124.7558	240.2007	359.0496	473.7550
20	53.0518	83.0743	163.0092	320.5970	479.2113	632.4806
40	66.3954	100.8658	204.4371	401.2863	598.2933	792.3785

Table 6: Polynomial degrees selected by Rissanen's classic ((k/2) log n) method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	21	2	2	6	11	58	0	42	6	7	3	6	13	23	78	8	6	4	0	1	3
5	19	3	7	9	6	56	0	45	2	8	4	8	10	23	78	11	4	1	4	2	0
10	10	5	5	5	12	63	0	39	5	5	5	8	11	27	78	9	5	2	1	2	3
20	19	3	3	4	16	55	0	39	11	9	3	3	10	25	86	7	1	1	3	1	1
40	26	5	4	2	10	53	0	33	12	4	4	5	10	32	78	8	7	0	3	1	3
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	91	5	3	0	1	0	0	96	3	0	1	0	0	0	94	6	0	0	0	0	0
5	93	6	1	0	0	0	0	94	4	1	0	0	1	0	95	4	1	0	0	0	0
10	94	6	0	0	0	0	0	91	8	1	0	0	0	0	94	3	2	0	1	0	0
20	95	2	2	0	0	0	1	93	5	2	0	0	0	0	96	3	1	0	0	0	0
40	91	6	1	0	2	0	0	95	3	1	0	1	0	0	92	8	0	0	0	0	0

Table 7: Polynomial degrees selected by Nohre's method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	99	1	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0
5	99	1	0	0	0	0	0	99	0	0	1	0	0	0	100	0	0	0	0	0	0
10	99	1	0	0	0	0	0	97	3	0	0	0	0	0	100	0	0	0	0	0	0
20	93	6	0	0	0	1	0	95	5	0	0	0	0	0	100	0	0	0	0	0	0
40	90	3	0	1	1	5	0	96	2	0	0	0	1	1	100	0	0	0	0	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	100	0	0	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0
5	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0
10	99	1	0	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0
20	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0
40	99	1	0	0	0	0	0	99	1	0	0	0	0	0	98	2	0	0	0	0	0

Table 8: Polynomial degrees selected by uniform prior/limited values method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0
5	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0
10	99	1	0	0	0	0	0	97	3	0	0	0	0	0	100	0	0	0	0	0	0
20	94	6	0	0	0	0	0	93	7	0	0	0	0	0	100	0	0	0	0	0	0
40	92	6	2	0	0	0	0	91	7	0	0	1	1	0	97	2	1	0	0	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	100	0	0	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0
5	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0
10	98	2	0	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0
20	100	0	0	0	0	0	0	99	1	0	0	0	0	0	99	1	0	0	0	0	0
40	95	4	1	0	0	0	0	97	3	0	0	0	0	0	95	5	0	0	0	0	0

Table 9: Polynomial degrees selected by uniform prior/unlimited values method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	99	1	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0
5	99	1	0	0	0	0	0	99	0	0	1	0	0	0	100	0	0	0	0	0	0
10	99	1	0	0	0	0	0	97	3	0	0	0	0	0	100	0	0	0	0	0	0
20	94	6	0	0	0	0	0	95	5	0	0	0	0	0	100	0	0	0	0	0	0
40	91	3	1	1	1	3	0	96	2	0	0	0	1	1	100	0	0	0	0	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	100	0	0	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0
5	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0
10	99	1	0	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0
20	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0
40	99	1	0	0	0	0	0	99	1	0	0	0	0	0	98	2	0	0	0	0	0

Table 10: Polynomial degrees selected by Maximum-Likelihood code method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma \downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	68	1	2	3	5	21	0	82	5	3	2	1	1	6	94	6	0	0	0	0	0
5	56	3	6	2	4	29	0	89	1	1	1	1	0	7	94	5	1	0	0	0	0
10	57	6	0	2	2	33	0	82	6	4	2	0	3	3	98	2	0	0	0	0	0
20	56	5	3	4	5	27	0	87	8	3	0	0	1	1	98	0	1	1	0	0	0
40	62	3	2	2	4	27	0	76	7	2	1	2	2	10	97	2	1	0	0	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma \downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	99	1	0	0	0	0	0	99	1	0	0	0	0	0	96	4	0	0	0	0	0
5	97	3	0	0	0	0	0	98	1	1	0	0	0	0	98	2	0	0	0	0	0
10	98	2	0	0	0	0	0	95	5	0	0	0	0	0	97	2	1	0	0	0	0
20	99	0	1	0	0	0	0	96	4	0	0	0	0	0	98	1	1	0	0	0	0
40	95	4	1	0	0	0	0	97	3	0	0	0	0	0	95	5	0	0	0	0	0

Table 11: Prediction error results for the code-length formulas investigated here. These results are for 7 data points.

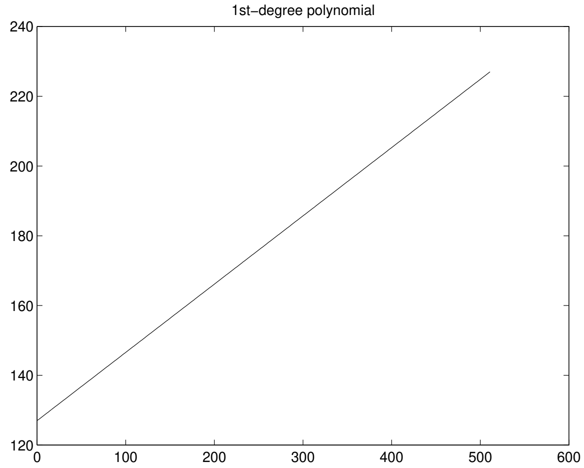
$\sigma \downarrow$ *method* →	Rissanen $(\frac{k}{2}\log n)$	Nohre	UP-UL	UP-LIM	ML
3	15.0286	0.8958	0.8958	0.8886	8.2361
5	33.4121	1.6195	1.6195	1.6014	19.8609
10	119.2872	3.0109	3.0109	3.0109	39.9395
20	138.1347	7.5651	7.2207	7.2207	68.5997
40	239.1173	32.5131	17.7251	14.0524	170.6604

Table 12: Polynomial degrees selected by various methods for the 7-point case.

degree ↘	Rissanen ($\frac{k}{2}\log n$)							Nohre						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	21	2	2	6	11	58	0	99	1	0	0	0	0	0
5	19	3	7	9	6	56	0	99	1	0	0	0	0	0
10	10	5	5	5	12	63	0	99	1	0	0	0	0	0
20	19	3	3	4	16	55	0	93	6	0	0	0	1	0
40	26	5	4	2	10	53	0	90	3	0	1	1	5	0

degree ↘	Uniform Prior (unlim.)							Uniform Prior (limited)							Maximum-Likelihood						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	99	1	0	0	0	0	0	100	0	0	0	0	0	0	68	1	2	3	5	21	0
5	99	1	0	0	0	0	0	100	0	0	0	0	0	0	56	3	6	2	4	29	0
10	99	1	0	0	0	0	0	99	1	0	0	0	0	0	57	6	0	2	2	33	0
20	94	6	0	0	0	0	0	94	6	0	0	0	0	0	56	5	3	4	5	27	0
40	91	3	1	1	1	3	0	92	6	2	0	0	0	0	62	3	2	2	4	27	0

D.2 1st-degree polynomial



(a)

Figure 5: First-degree polynomial used in curve fitting experiments

Experimental parameters:

- Polynomial Coefficients: The results in this subsection are for a first-degree polynomial of the form $y = C_1 x \; + \; C_0$ using the following values.

$$C_1 = (100/511) = 0.1956947$$
$$C_0 = 127$$

  A graph of this function appears in Figure 5.

- (xmin, xmax, ymin, ymax) = [0, 511, 0, 255]
- number of trials at each (σ, n_p) pair = 100
- σ values = 3, 5, 10, 20, 40
- numbers of points (n_p) = 7, 10, 20, 40, 60, 80

Table 13: Prediction error results for the five code-length formulas investigated here.

$\sigma \downarrow \; n_p \rightarrow$	7	10	20	40	60	80
			$\frac{k}{2}\log n$ (Rissanen)			
3	52.7742	4.5719	1.0188	0.6587	0.5601	0.4374
5	91.9894	7.5472	1.5591	1.0972	0.8498	0.6687
10	50.1802	17.6057	3.5678	2.0658	1.7562	1.4664
20	82.7456	52.2464	6.1923	4.4435	3.4790	2.8985
40	601.4315	72.2715	13.9132	8.0783	6.1177	6.1262
			Nohre			
3	1.3419	1.1987	0.8084	0.6011	0.4889	0.3948
5	2.5344	1.8052	1.3231	1.0067	0.7917	0.6456
10	4.6450	3.9976	2.7567	1.9723	1.6719	1.3769
20	10.9631	8.4224	5.1511	4.0296	3.2161	2.6166
40	39.2898	28.0764	14.9679	7.6493	5.7472	5.8905
			Uniform Prior (limited)			
3	1.3174	1.1987	0.8084	0.6011	0.4889	0.3948
5	2.1675	1.8052	1.3231	1.0067	0.7917	0.6456
10	4.1716	3.9976	2.8287	1.9874	1.6719	1.3769
20	9.5635	7.2238	5.2179	4.0296	3.2161	2.6708
40	26.1241	22.2202	12.5788	7.4706	5.8044	5.8905
			Uniform Prior (unlimited)			
3	1.3453	1.1987	0.8084	0.6011	0.4889	0.3948
5	2.2158	1.8052	1.3231	1.0067	0.7917	0.6456
10	4.2184	3.9976	2.7567	1.9723	1.6719	1.3769
20	10.9022	8.4224	5.1511	4.0296	3.2161	2.6166
40	39.3356	28.0764	14.9679	7.6493	5.7472	5.8905
			Maximum-Likelihood			
3	44.3436	2.1802	0.8616	0.6150	0.5092	0.3983
5	84.6724	4.9260	1.3388	1.0644	0.8233	0.6687
10	38.4865	5.5897	3.2829	1.9958	1.6991	1.3995
20	58.8625	26.1480	5.6321	4.1276	3.2161	2.7382
40	141.0723	28.2982	13.0424	7.4706	5.8044	5.8905

Table 14: Total encoding length (in bits; rounded to nearest bit) obtained using the encoding schemes investigated here.

$\sigma \downarrow \;\; n_p \rightarrow$	7	10	20	40	60	80
$\frac{k}{2}\log n$ (Rissanen)						
3	19.0094	33.9276	71.5987	142.3966	213.5672	280.6204
5	20.9858	40.5497	86.6690	172.3268	255.6632	338.4840
10	27.3520	51.0589	107.0483	211.3202	315.7836	420.2948
20	34.9478	60.5410	127.1314	249.8482	375.1918	498.7795
40	42.9143	70.3006	144.4638	288.8711	431.4492	574.2913
Nohre						
3	55.4026	67.9820	106.1290	181.6554	257.5429	329.2226
5	59.8283	73.8190	119.9216	210.5339	298.5377	385.9815
10	64.8081	82.5484	139.2032	247.9250	357.0408	466.1609
20	69.8686	90.8699	157.3567	284.8779	414.8317	543.0422
40	73.9319	97.5265	172.6696	322.1427	469.4232	616.8320
Uniform Prior (limited)						
3	42.2524	53.9285	91.0636	165.7945	241.2945	312.8133
5	45.3037	58.6306	103.6827	193.5367	281.2398	368.4951
10	48.8653	65.7669	121.4133	229.5417	338.3391	447.2369
20	52.5889	72.7593	138.2321	265.2079	394.8020	522.8199
40	56.1152	78.9627	152.7784	301.2825	448.2131	595.4302
Uniform Prior (unlimited)						
3	52.6649	64.9203	102.7221	178.1265	254.0328	325.7133
5	57.1970	70.8431	116.6703	207.2082	295.1758	382.5999
10	62.3539	79.8236	136.2217	244.7779	353.8283	462.9097
20	67.6123	88.3702	154.5404	281.8551	411.7261	539.8840
40	71.8249	95.1413	169.9558	319.2147	466.3961	613.7446
Maximum-Likelihood						
3	19.8195	33.1646	58.8890	108.9765	160.9529	204.8232
5	24.6793	47.2219	88.6452	168.8371	245.0397	320.5207
10	37.3654	67.3934	129.9400	246.7610	365.2466	484.1343
20	53.3018	86.6928	169.8835	323.9164	484.1410	641.1717
40	67.2408	104.7783	204.1202	401.9327	596.6068	792.1069

Table 15: Polynomial degrees selected by Rissanen's classic ((k/2) log n) method.

degree ↘	np = 7							np = 10							np = 20						
σ ↓	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	16	7	5	14	58	0	0	40	13	6	6	9	26	0	77	11	7	2	2	1
5	0	13	6	8	8	65	0	0	36	9	9	8	10	28	0	87	7	2	2	1	1
10	0	14	2	11	11	62	0	0	42	9	4	7	12	26	0	78	12	6	2	2	0
20	0	10	6	10	9	65	0	0	37	6	2	7	13	35	0	83	6	4	3	2	2
40	2	14	3	6	16	59	0	11	32	5	7	10	6	29	5	79	9	3	0	1	3
degree ↘	np = 40							np = 60							np = 80						
σ ↓	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	93	4	1	0	2	0	0	90	2	7	1	0	0	0	94	2	1	2	1	0
5	0	94	4	2	0	0	0	0	93	7	0	0	0	0	0	98	0	2	0	0	0
10	0	95	2	2	1	0	0	0	94	5	1	0	0	0	0	96	3	0	1	0	0
20	0	92	5	1	0	2	0	0	93	5	1	0	1	0	0	92	7	1	0	0	0
40	0	93	3	3	0	1	0	0	94	5	0	1	0	0	0	95	5	0	0	0	0

Table 16: Polynomial degrees selected by Nohre's method.

degree ↘	np = 7							np = 10							np = 20						
σ ↓	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	98	2	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
5	0	98	1	0	1	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
10	0	97	1	1	1	0	0	0	98	2	0	0	0	0	0	99	1	0	0	0	0
20	9	85	4	1	1	0	0	5	91	3	0	0	0	1	0	99	1	0	0	0	0
40	60	34	2	1	1	2	0	63	34	1	1	0	1	0	24	75	1	0	0	0	0
degree ↘	np = 40							np = 60							np = 80						
σ ↓	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
5	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
10	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
20	0	100	0	0	0	0	0	0	99	1	0	0	0	0	0	99	1	0	0	0	0
40	3	97	0	0	0	0	0	0	99	1	0	0	0	0	0	99	1	0	0	0	0

Table 17: Polynomial degrees selected by uniform-prior/limited-values method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
5	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
10	0	100	0	0	0	0	0	0	98	2	0	0	0	0	0	97	3	0	0	0	0
20	3	92	4	1	0	0	0	0	96	4	0	0	0	0	0	98	2	0	0	0	0
40	41	52	4	3	0	0	0	35	60	3	1	0	1	0	13	85	2	0	0	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
5	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
10	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
20	0	100	0	0	0	0	0	0	99	1	0	0	0	0	0	98	2	0	0	0	0
40	1	98	1	0	0	0	0	0	98	2	0	0	0	0	0	99	1	0	0	0	0

Table 18: Polynomial degrees selected by uniform prior/unlimited values method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	97	3	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
5	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
10	0	98	1	1	0	0	0	0	98	2	0	0	0	0	0	99	1	0	0	0	0
20	9	86	4	1	0	0	0	5	91	3	0	0	0	1	0	99	1	0	0	0	0
40	61	34	2	1	1	1	0	63	34	1	1	0	1	0	24	75	1	0	0	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
5	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
10	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0
20	0	100	0	0	0	0	0	0	99	1	0	0	0	0	0	99	1	0	0	0	0
40	3	97	0	0	0	0	0	0	99	1	0	0	0	0	0	99	1	0	0	0	0

Table 19: Polynomial degrees selected by Maximum-Likelihood-code method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	54	7	4	5	30	0	0	82	6	2	2	3	5	0	93	4	2	1	0	0
5	0	49	2	4	7	38	0	0	70	7	7	2	4	10	0	99	1	0	0	0	0
10	0	46	1	5	7	41	0	0	79	3	6	3	3	6	0	86	9	4	1	0	0
20	0	42	6	5	3	44	0	0	82	4	0	2	2	10	0	92	4	3	0	1	0
40	25	31	3	3	5	33	0	35	49	2	2	2	4	6	15	83	2	0	0	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	98	1	1	0	0	0	0	97	2	1	0	0	0	0	99	1	0	0	0	0
5	0	97	2	1	0	0	0	0	97	3	0	0	0	0	0	98	0	2	0	0	0
10	0	98	2	0	0	0	0	0	98	2	0	0	0	0	0	99	1	0	0	0	0
20	0	98	2	0	0	0	0	0	99	1	0	0	0	0	0	96	4	0	0	0	0
40	1	98	1	0	0	0	0	0	98	2	0	0	0	0	0	99	1	0	0	0	0

Table 20: Prediction error results for the code-length formulas investigated here. These results are for 7 data points.

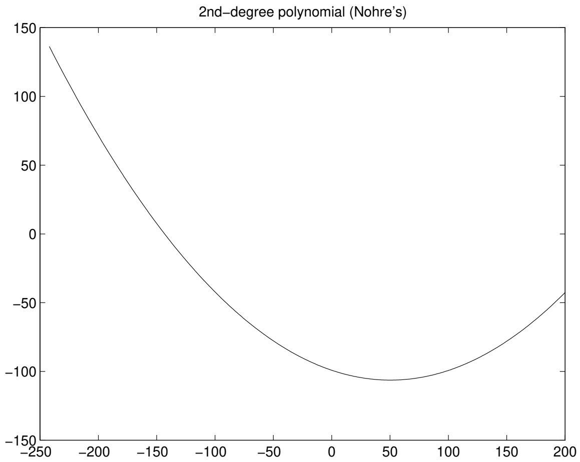
$\sigma\downarrow$ *method* →	Rissanen ($\frac{k}{2}\log n$)	Nohre	UP-UL	UP-LIM	ML
3	52.7742	1.3419	1.3453	1.3174	44.3436
5	91.9894	2.5344	2.2158	2.1675	84.6724
10	50.1802	4.6450	4.2184	4.1716	38.4865
20	82.7456	10.9631	10.9022	9.5635	58.8625
40	601.4315	39.2898	39.3356	26.1241	141.0723

Table 21: Polynomial degrees selected by various methods for the 7-point case.

degree ↘	Rissanen ($\frac{k}{2}\log n$)							Nohre						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	16	7	5	14	58	0	0	98	2	0	0	0	0
5	0	13	6	8	8	65	0	0	98	1	0	1	0	0
10	0	14	2	11	11	62	0	0	97	1	1	1	0	0
20	0	10	6	10	9	65	0	9	85	4	1	1	0	0
40	2	14	3	6	16	59	0	60	34	2	1	1	2	0

degree ↘	Uniform Prior (unlim.)							Uniform Prior (limited)							Maximum-Likelihood						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	97	3	0	0	0	0	0	100	0	0	0	0	0	0	54	7	4	5	30	0
5	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	49	2	4	7	38	0
10	0	98	1	1	0	0	0	0	100	0	0	0	0	0	0	46	1	5	7	41	0
20	9	86	4	1	0	0	0	3	92	4	1	0	0	0	0	42	6	5	3	44	0
40	61	34	2	1	1	1	0	41	52	4	3	0	0	0	25	31	3	3	5	33	0

D.3 2nd-degree polynomial (Nohre's)



(a)

Figure 6: Second-degree polynomial used in curve fitting experiments. This is the same polynomial for which results were reported in [12].

Experimental parameters:

- Polynomial Coefficients: The results in this subsection are for a second-degree polynomial of the form $y = C_2x^2 + C_1x + C_0$ using the following values.

  $$C_2 = 0.002840182$$
  $$C_1 = -0.2854313$$
  $$C_0 = -99.11219$$

  A graph of this function appears in Figure 6. This is the same polynomial for which results were reported in [12].

- (xmin, xmax, ymin, ymax) = [0, 511, 0, 255]
- number of trials at each (σ, n_p) pair = 100
- σ values = 3, 5, 10, 20, 40
- numbers of points (n_p) = 7, 10, 20, 40, 60, 80

Table 22: Prediction error results for the five code-length formulas investigated here.

$\sigma \downarrow \ n_p \rightarrow$	7	10	20	40	60	80
$\frac{k}{2}\log n$ (Rissanen)						
3	26.0353	10.1129	1.3351	0.8715	0.6832	0.5487
5	24.1208	9.6496	2.4987	1.3129	1.1333	0.8949
10	65.4654	18.8983	3.7171	2.8059	2.2222	1.8413
20	200.5477	22.0454	8.3216	5.4536	4.4471	3.9510
40	233.2479	81.4801	16.6584	11.2928	8.6327	7.0955
Nohre						
3	1.7420	1.5173	1.0961	0.7783	0.6223	0.5078
5	3.7917	2.2692	1.6917	1.1808	1.0551	0.8740
10	7.7640	5.1681	3.1244	2.6373	2.1147	1.7697
20	128.5057	10.5525	6.9196	5.1657	4.2107	3.7770
40	160.9505	43.6533	14.4804	10.7400	8.4171	6.9656
Uniform Prior (limited)						
3	1.6738	1.5210	1.0961	0.7783	0.6223	0.5078
5	2.9749	2.2489	1.6917	1.1808	1.0420	0.8740
10	6.2162	5.1281	3.1054	2.6373	2.1147	1.7697
20	12.9507	9.4925	6.7314	5.1657	4.1299	3.7071
40	28.0298	18.9391	13.7594	10.5135	8.2701	6.9250
Uniform Prior (unlimited)						
3	1.7348	1.5173	1.0961	0.7783	0.6223	0.5078
5	3.1256	2.2692	1.6917	1.1808	1.0551	0.8740
10	7.3170	5.1681	3.1244	2.6373	2.1147	1.7697
20	112.3571	10.5525	6.9196	5.1657	4.2107	3.7770
40	87.4595	43.8551	14.4804	10.7400	8.4171	6.9656
Maximum-Likelihood						
3	9.3982	6.3082	1.1411	0.7844	0.6313	0.5215
5	18.3507	4.1464	1.7493	1.1808	1.0728	0.8814
10	43.9266	12.9325	3.2709	2.6373	2.1147	1.7894
20	138.5692	13.1315	7.0010	5.1657	4.2107	3.7770
40	210.3525	45.7414	14.0504	10.5135	8.2065	6.9250

Table 23: Total encoding length (in bits; rounded to nearest bit) obtained using the encoding schemes investigated here.

$\sigma \downarrow$ $n_p \rightarrow$	7	10	20	40	60	80
			$\frac{k}{2}\log n$ (Rissanen)			
3	17.2770	35.3415	73.4377	143.3768	214.0047	283.5640
5	21.7675	41.7325	87.4980	174.3201	258.3009	341.9074
10	30.1679	51.3984	107.9403	212.0731	317.6235	419.6657
20	35.1205	61.1350	127.9042	253.6698	377.0141	500.4763
40	42.5517	71.1138	147.2980	291.8933	437.8658	580.7171
			Nohre			
3	57.6494	71.4924	110.2437	185.0323	260.3047	334.4820
5	60.8202	76.3611	122.6894	214.0289	302.7627	390.8856
10	65.6923	83.6529	140.5261	249.3989	359.4947	466.1646
20	68.7389	90.8737	158.0132	288.2525	416.3035	544.2941
40	73.2446	97.8243	174.6716	323.7664	474.3146	621.7780
			Uniform Prior (limited)			
3	49.7221	62.2896	100.0456	174.0433	248.9373	322.9063
5	51.7918	66.0742	111.3437	201.8901	290.3034	378.2403
10	55.0437	71.8592	127.6933	235.8924	345.6545	452.1785
20	57.3334	77.7213	143.7375	273.3533	401.1528	528.9460
40	60.3884	83.8653	159.2399	307.6779	457.8536	605.1355
			Uniform Prior (unlimited)			
3	54.9891	68.5164	106.8579	181.4980	256.7868	330.9739
5	58.4209	73.4674	119.4419	210.7057	299.4034	387.5065
10	63.4092	80.9790	137.5526	246.2521	356.2841	462.9108
20	66.8068	88.4262	155.2123	285.2411	413.2007	541.1372
40	71.4618	95.5553	171.9898	320.8476	471.2966	618.6966
			Maximum-Likelihood			
3	21.5563	41.5401	67.6285	115.6944	166.2687	214.9912
5	30.6205	54.3717	95.8552	177.5819	254.8261	331.5812
10	47.4012	73.4745	136.7631	253.0967	373.5247	487.1675
20	56.7079	92.8710	176.8667	336.3407	492.3757	648.8089
40	72.4159	112.9651	215.6864	412.6657	613.9533	809.2506

Table 24: Polynomial degrees selected by Rissanen's classic ((k/2) log n) method.

degree ↘	np = 7							np = 10							np = 20						
σ ↓	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	21	6	12	61	0	0	0	40	12	6	7	35	0	0	75	16	1	6	2
5	0	0	14	10	9	67	0	0	0	36	9	12	13	30	0	0	79	11	3	4	3
10	0	0	15	7	8	70	0	0	0	41	12	7	12	28	0	0	81	7	5	5	2
20	0	0	20	10	14	56	0	0	0	47	8	7	7	31	0	0	71	14	7	7	1
40	0	0	15	4	20	61	0	0	2	32	11	11	12	32	0	0	76	9	6	4	5
degree ↘	np = 40							np = 60							np = 80						
σ ↓	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	87	7	4	1	1	0	0	87	10	1	2	0	0	0	93	5	1	0	1
5	0	0	89	5	3	2	1	0	0	91	7	1	0	1	0	0	97	3	0	0	0
10	0	0	91	5	4	0	0	0	0	95	1	2	2	0	0	0	94	5	1	0	0
20	0	0	94	4	0	1	1	0	0	93	3	2	2	0	0	0	92	6	1	1	0
40	0	0	91	5	2	2	0	0	0	93	6	1	0	0	0	0	94	6	0	0	0

Table 25: Polynomial degrees selected by Nohre's method.

degree ↘	np = 7							np = 10							np = 20						
σ ↓	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	96	3	1	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0
5	0	0	95	1	3	1	0	0	0	98	2	0	0	0	0	0	99	1	0	0	0
10	1	0	90	6	2	1	0	0	0	98	1	1	0	0	0	0	99	1	0	0	0
20	2	0	72	8	7	11	0	0	0	95	2	2	0	1	0	0	96	4	0	0	0
40	12	5	43	8	8	24	0	7	2	75	6	1	6	3	0	0	90	7	2	0	1
degree ↘	np = 40							np = 60							np = 80						
σ ↓	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0
5	0	0	100	0	0	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0
10	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	99	1	0	0	0
20	0	0	98	2	0	0	0	0	0	97	2	1	0	0	0	0	97	3	0	0	0
40	0	0	98	0	1	1	0	0	0	97	2	1	0	0	0	0	97	3	0	0	0

Table 26: Polynomial degrees selected by uniform-prior/limited-values method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0
5	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	99	1	0	0	0
10	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0
20	1	0	98	1	0	0	0	0	0	100	0	0	0	0	0	0	99	1	0	0	0
40	4	14	80	2	0	0	0	0	6	92	2	0	0	0	0	2	96	2	0	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0
5	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0
10	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	99	1	0	0	0
20	0	0	98	2	0	0	0	0	0	99	0	1	0	0	0	0	99	1	0	0	0
40	0	0	100	0	0	0	0	0	0	99	1	0	0	0	0	0	98	2	0	0	0

Table 27: Polynomial degrees selected by uniform-prior/unlimited-values method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	97	3	0	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0
5	0	0	95	5	0	0	0	0	0	98	2	0	0	0	0	0	99	1	0	0	0
10	1	0	92	6	1	0	0	0	0	98	1	1	0	0	0	0	99	1	0	0	0
20	3	0	76	8	6	7	0	0	0	95	2	2	0	1	0	0	96	4	0	0	0
40	17	5	52	9	8	9	0	7	3	74	6	1	6	3	0	0	90	7	2	0	1
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0
5	0	0	100	0	0	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0
10	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	99	1	0	0	0
20	0	0	98	2	0	0	0	0	0	97	2	1	0	0	0	0	97	3	0	0	0
40	0	0	98	0	1	1	0	0	0	97	2	1	0	0	0	0	97	3	0	0	0

Table 28: Polynomial degrees selected by Maximum-Likelihood-code method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma \downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	51	7	6	36	0	0	0	74	9	6	1	10	0	0	93	7	0	0	0
5	0	0	48	5	4	43	0	0	0	78	7	6	4	5	0	0	95	5	0	0	0
10	0	0	50	6	6	38	0	0	0	81	4	3	5	7	0	0	96	3	0	1	0
20	0	0	53	4	12	31	0	0	0	83	3	2	1	11	0	0	95	4	0	1	0
40	3	6	30	5	12	44	0	0	5	73	6	1	6	9	0	2	93	3	1	0	1
degree ↘	np = 40							np = 60							np = 80						
$\sigma \downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	99	1	0	0	0	0	0	98	2	0	0	0	0	0	97	3	0	0	0
5	0	0	100	0	0	0	0	0	0	97	3	0	0	0	0	0	99	1	0	0	0
10	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	98	2	0	0	0
20	0	0	98	2	0	0	0	0	0	97	2	1	0	0	0	0	97	3	0	0	0
40	0	0	100	0	0	0	0	0	0	99	1	0	0	0	0	0	98	2	0	0	0

Table 29: Prediction error results for the code-length formulas investigated here. These results are for 7 data points.

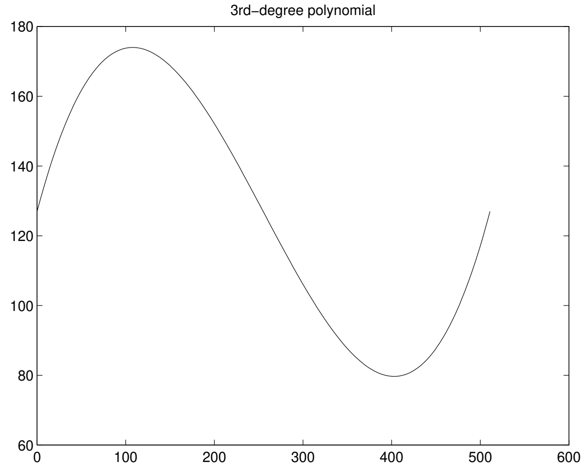
$\sigma \downarrow$ *method* →	Rissanen ($\frac{k}{2}\log n$)	Nohre	UP-UL	UP-LIM	ML
3	26.0353	1.7420	1.7348	1.6738	9.3982
5	24.1208	3.7917	3.1256	2.9749	18.3507
10	65.4654	7.7640	7.3170	6.2162	43.9266
20	200.5477	128.5057	112.3571	12.9507	138.5692
40	233.2479	160.9505	87.4595	28.0298	210.3525

Table 30: Polynomial degrees selected by various methods for the 7-point case.

degree ↘	Rissanen ($\frac{k}{2}\log n$)							Nohre						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	21	6	12	61	0	0	0	96	3	1	0	0
5	0	0	14	10	9	67	0	0	0	95	1	3	1	0
10	0	0	15	7	8	70	0	1	0	90	6	2	1	0
20	0	0	20	10	14	56	0	2	0	72	8	7	11	0
40	0	0	15	4	20	61	0	12	5	43	8	8	24	0

degree ↘	Uniform Prior (unlim.)							Uniform Prior (limited)							Maximum-Likelihood						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	97	3	0	0	0	0	0	100	0	0	0	0	0	0	51	7	6	36	0
5	0	0	95	5	0	0	0	0	0	100	0	0	0	0	0	0	48	5	4	43	0
10	1	0	92	6	1	0	0	0	0	99	1	0	0	0	0	0	50	6	6	38	0
20	3	0	76	8	6	7	0	1	0	98	1	0	0	0	0	0	53	4	12	31	0
40	17	5	52	9	8	9	0	4	14	80	2	0	0	0	3	6	30	5	12	44	0

D.4 3rd-degree polynomial



(a)

Figure 7: Third-degree polynomial used in curve fitting experiments.

Experimental parameters:

- Polynomial Coefficients: The results in this subsection are for a third-degree polynomial of the form $y = C_3x^3 + C_2x^2 + C_1x + C_0$ using the following values.

  $C_3 = 7.343750 \times 10^{-6}$

  $C_2 = -5.625313 \times 10^{-3}$

  $C_1 = 0.9569273$

  $C_0 = 127$

  A graph of this function appears in Figure 7.

- (xmin, xmax, ymin, ymax) = [0, 511, 0, 255]

- number of trials at each (σ, n_p) pair = 100

- σ values = 3, 5, 10, 20, 40

- numbers of points (n_p) = 7, 10, 20, 40, 60, 80

Table 31: Prediction error results for the five code-length formulas investigated here.

$\sigma \downarrow \;\; n_p \rightarrow$	7	10	20	40	60	80
$\frac{k}{2} \log n$ (Rissanen)						
3	54.1058	6.3625	1.4209	0.9038	0.8161	0.6509
5	634.1060	9.8455	2.2574	1.5919	1.2407	1.0433
10	52.0924	19.5424	4.8278	2.9575	2.4637	2.1102
20	81.3675	84.4252	8.8348	6.2188	4.8472	4.0791
40	491.7290	73.9062	22.9215	14.4231	10.7712	10.0145
Nohre						
3	5.9430	1.9094	1.1905	0.8330	0.7611	0.6276
5	15.4731	3.2152	1.8755	1.4580	1.2059	0.9956
10	36.6669	16.2174	4.2578	2.8500	2.3983	2.0266
20	35.3903	31.6349	13.6626	5.9116	4.7924	3.9844
40	55.3429	39.4844	29.7833	18.4702	13.8937	11.4100
Uniform Prior (limited)						
3	15.9212	1.8883	1.1905	0.8330	0.7611	0.6276
5	21.7264	3.1306	1.8755	1.4580	1.2059	0.9956
10	31.5023	12.2094	4.2578	2.8500	2.3983	2.0266
20	33.0230	29.0457	10.6082	5.9500	4.7924	3.9844
40	36.6861	37.0886	26.3015	16.3770	12.3482	10.3551
Uniform Prior (unlimited)						
3	5.2916	1.9069	1.1905	0.8330	0.7611	0.6276
5	14.9351	3.2152	1.8755	1.4580	1.2059	0.9956
10	35.6826	15.9066	4.2578	2.8500	2.3983	2.0266
20	32.7867	31.6349	13.6626	5.9116	4.7924	3.9844
40	51.6372	38.3461	29.7833	18.4702	13.8937	11.4100
Maximum-Likelihood						
3	52.1424	4.6585	1.2856	0.8536	0.7692	0.6454
5	90.5699	7.0919	1.9994	1.4943	1.2248	1.0143
10	45.6046	13.1490	4.4211	2.9390	2.4200	2.0752
20	71.6914	43.6577	9.6196	6.1502	4.7924	4.0311
40	116.9943	41.1275	27.0946	16.7766	12.6995	10.7321

Table 32: Total encoding length (in bits; rounded to nearest bit) obtained using the encoding schemes investigated here.

$\sigma \downarrow$ $n_p \rightarrow$	7	10	20	40	60	80
$\frac{k}{2}\log n$ (Rissanen)						
3	19.1726	34.6680	73.8384	146.5567	217.7224	285.4031
5	22.2794	41.2494	89.3223	175.9979	259.9408	343.3344
10	27.7399	51.9779	109.5952	215.2141	320.2616	425.1973
20	35.0625	61.2306	129.8645	253.8800	380.4752	504.3700
40	43.7021	71.1660	147.6053	294.3600	439.0111	583.1713
Nohre						
3	64.4572	78.5857	118.2657	195.4120	271.2102	343.7134
5	66.6341	82.8083	131.0481	222.3239	310.9294	398.9650
10	68.2535	89.1055	147.5770	257.9938	367.5964	477.1485
20	70.2890	92.6052	163.3989	293.1930	424.1840	552.6217
40	73.2901	97.8495	175.1541	328.0808	478.0491	627.1120
Uniform Prior (limited)						
3	51.7404	64.5516	102.6666	178.9216	254.3243	326.5977
5	52.6641	67.6790	114.2113	204.7028	292.9603	380.8013
10	53.0652	72.5822	129.3299	238.9963	348.2172	457.5628
20	54.3824	75.8260	144.2254	272.8750	403.4735	531.7301
40	56.5989	80.2873	156.2248	307.4553	456.6672	605.3259
Uniform Prior (unlimited)						
3	62.1406	75.7146	114.8955	191.8878	267.6983	340.2034
5	64.5331	80.0154	127.8231	219.0030	307.5696	395.5849
10	66.0587	86.5329	144.6231	254.8570	364.3892	473.9008
20	68.1798	90.1555	160.6220	290.1834	421.0864	549.4687
40	71.2261	95.4788	172.4612	325.1677	475.0332	624.0330
Maximum-Likelihood						
3	23.5526	40.4130	69.7487	122.8573	174.2656	219.0485
5	30.1607	53.5492	100.5719	181.7750	258.5753	334.8421
10	40.7693	75.1164	140.9462	260.1458	379.1631	498.5887
20	54.4225	92.1084	181.1603	337.6024	499.6647	656.8716
40	68.9674	107.5935	212.1197	415.9544	615.3260	813.6327

Table 33: Polynomial degrees selected by Rissanen's classic ((k/2) log n) method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	21	19	60	0	0	0	0	43	11	10	36	0	0	0	74	10	11	5
5	0	0	0	18	15	67	0	0	0	0	33	17	20	30	0	0	0	77	11	9	3
10	0	0	1	19	17	63	0	0	0	0	37	14	17	32	0	0	0	78	12	8	2
20	2	1	0	17	14	66	0	3	2	1	21	10	19	44	1	0	0	84	10	2	3
40	9	6	2	8	12	63	0	15	10	3	20	10	9	33	15	19	2	51	4	3	6
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	90	4	5	1	0	0	0	90	6	3	1	0	0	0	93	5	1	1
5	0	0	0	89	6	3	2	0	0	0	95	5	0	0	0	0	0	94	3	3	0
10	0	0	0	93	6	1	0	0	0	0	96	3	1	0	0	0	0	93	6	1	0
20	0	0	0	91	4	5	0	0	0	0	96	2	2	0	0	0	0	97	2	1	0
40	0	28	2	66	3	0	1	0	14	1	78	7	0	0	1	13	0	82	3	1	0

Table 34: Polynomial degrees selected by Nohre's method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	10	0	0	85	5	0	0	0	0	0	98	1	1	0	0	0	0	100	0	0	0
5	32	0	0	55	11	2	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0
10	65	1	1	27	3	3	0	34	2	0	63	0	0	1	0	0	0	99	1	0	0
20	74	6	2	9	3	6	0	76	8	2	11	1	0	2	9	26	0	64	0	1	0
40	84	6	0	2	2	6	0	83	8	1	4	1	1	2	61	25	0	13	1	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	99	1	0	0
5	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0
10	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	99	1	0	0
20	0	0	0	99	1	0	0	0	0	0	98	1	1	0	0	0	0	100	0	0	0
40	11	50	0	38	1	0	0	2	41	0	56	1	0	0	1	26	0	73	0	0	0

Table 35: Polynomial degrees selected by uniform prior/limited values method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	39	0	1	60	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0
5	55	1	0	44	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0
10	84	3	0	13	0	0	0	19	5	0	76	0	0	0	0	0	0	99	1	0	0
20	81	9	2	8	0	0	0	64	12	2	20	1	0	1	2	17	0	80	0	1	0
40	85	11	0	4	0	0	0	75	10	2	11	1	1	0	40	35	0	24	1	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	99	1	0	0
5	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0
10	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	99	1	0	0
20	0	0	0	98	2	0	0	0	0	0	98	1	1	0	0	0	0	100	0	0	0
40	3	44	2	50	1	0	0	1	29	1	68	1	0	0	1	18	0	81	0	0	0

Table 36: Polynomial degrees selected by uniform prior/unlimited values method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	8	0	0	88	4	0	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0
5	32	0	0	61	7	0	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0
10	67	1	1	27	3	1	0	33	2	0	65	0	0	0	0	0	0	99	1	0	0
20	79	7	2	9	2	1	0	76	8	2	11	1	0	2	9	26	0	64	0	1	0
40	89	6	0	2	1	2	0	84	8	1	4	1	1	1	61	25	0	13	1	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	99	1	0	0
5	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0
10	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	99	1	0	0
20	0	0	0	99	1	0	0	0	0	0	98	1	1	0	0	0	0	100	0	0	0
40	11	50	0	38	1	0	0	2	41	0	56	1	0	0	1	26	0	73	0	0	0

Table 37: Polynomial degrees selected by Maximum-Likelihood code method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma \downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	43	12	45	0	0	0	0	64	12	7	17	0	0	0	91	5	4	0
5	0	0	1	31	17	51	0	0	0	0	65	8	10	17	0	0	0	94	4	1	1
10	4	0	0	30	14	52	0	2	0	0	65	9	12	12	0	0	0	94	5	1	0
20	18	1	0	21	5	55	0	28	12	1	30	7	9	13	2	8	0	87	0	3	0
40	42	9	1	7	7	34	0	67	6	0	11	3	6	7	45	32	0	22	1	0	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma \downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	97	2	1	0	0	0	0	98	2	0	0	0	0	0	94	5	1	0
5	0	0	0	97	3	0	0	0	0	0	98	2	0	0	0	0	0	98	1	1	0
10	0	0	0	95	4	1	0	0	0	0	99	1	0	0	0	0	0	96	4	0	0
20	0	0	0	93	3	4	0	0	0	0	98	1	1	0	0	0	0	99	0	1	0
40	4	46	2	47	1	0	0	1	33	0	65	1	0	0	1	21	0	78	0	0	0

Table 38: Prediction error results for the code-length formulas investigated here. These results are for 7 data points.

$\sigma \downarrow$ *method* →	Rissanen ($\frac{k}{2}\log n$)	Nohre	UP-UL	UP-LIM	ML
3	54.1058	5.9430	5.2916	15.9212	52.1424
5	634.1060	15.4731	14.9351	21.7264	90.5699
10	52.0924	36.6669	35.6826	31.5023	45.6046
20	81.3675	35.3903	32.7867	33.0230	71.6914
40	491.7290	55.3429	51.6372	36.6861	116.9943

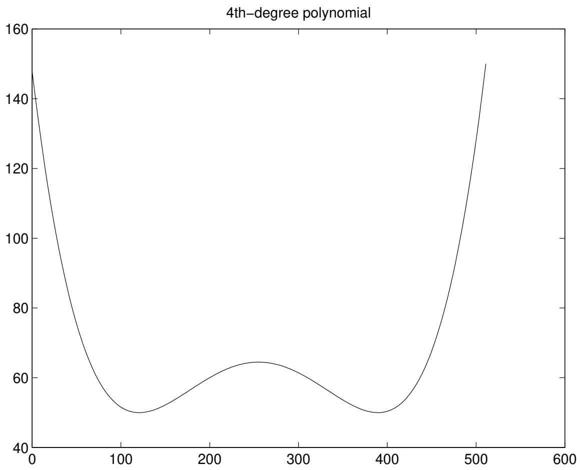
Table 39: Polynomial degrees selected by various methods for the 7-point case.

degree ↘	Rissanen ($\frac{k}{2}\log n$)							Nohre						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	21	19	60	0	10	0	0	85	5	0	0
5	0	0	0	18	15	67	0	32	0	0	55	11	2	0
10	0	0	1	19	17	63	0	65	1	1	27	3	3	0
20	2	1	0	17	14	66	0	74	6	2	9	3	6	0
40	9	6	2	8	12	63	0	84	6	0	2	2	6	0

degree ↘	Uniform Prior (unlim.)							Uniform Prior (limited)							Maximum-Likelihood						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	8	0	0	88	4	0	0	39	0	1	60	0	0	0	0	0	0	43	12	45	0
5	32	0	0	61	7	0	0	55	1	0	44	0	0	0	0	0	1	31	17	51	0
10	67	1	1	27	3	1	0	84	3	0	13	0	0	0	4	0	0	30	14	52	0
20	79	7	2	9	2	1	0	81	9	2	8	0	0	0	18	1	0	21	5	55	0
40	89	6	0	2	1	2	0	85	11	0	4	0	0	0	42	9	1	7	7	34	0

D.5 4th-degree polynomial

D.5.1 Results Covering a Broad Range of Numbers of Points



(a)

Figure 8: Fourth-degree polynomial used in curve fitting experiments.

Experimental parameters:

- Polynomial Coefficients: The results in this subsection are for a fourth-degree polynomial of the form $y = C_4x^4 + C_3x^3 + C_2x^2 + C_1x + C_0$ using the following values.

  $C_4 = 4.437105 \times 10^{-8}$

  $C_3 = -4.525847 \times 10^{-5}$

  $C_2 = 0.01570846$

  $C_1 = -2.125451$

  $C_0 = 147.8588$

  A graph of this function appears in Figure 8.

- (xmin, xmax, ymin, ymax) = [0, 511, 0, 255]

- number of trials at each (σ, n_p) pair = 100

- σ values = 3, 5, 10, 20, 40

- numbers of points (n_p) = 7, 10, 20, 40, 60, 80

Table 40: Prediction error results for the five code-length formulas investigated here.

$\sigma \downarrow$ $n_p \rightarrow$	7	10	20	40	60	80
$\frac{k}{2} \log n$ (Rissanen)						
3	14.8900	5.4431	1.4766	1.0154	0.8363	0.6852
5	104.4036	42.2322	2.9863	1.6777	1.3350	1.1847
10	15428.2551	36.2137	5.2930	3.5178	2.8705	2.3677
20	442.1098	63.6429	11.2708	6.9579	5.4901	4.8446
40	188.6660	153.8735	23.4656	16.6383	14.0344	11.8581
Nohre						
3	8.3955	2.9906	1.3668	0.9785	0.8134	0.6605
5	64.1869	7.4353	2.4118	1.6412	1.3024	1.1594
10	27.1485	11.2580	4.9692	3.4495	2.8239	2.3510
20	55.9534	25.0559	14.1918	7.2049	5.5516	4.7808
40	94.3797	32.7649	25.4009	19.4999	15.9778	12.8562
Uniform Prior (limited)						
3	24.2774	2.9742	1.3668	0.9785	0.8134	0.6605
5	26.8255	7.1674	2.4118	1.6412	1.3024	1.1594
10	28.8380	15.3394	4.9438	3.4400	2.8239	2.3510
20	32.3318	23.8503	14.3694	7.2829	5.5218	4.7603
40	36.4271	32.1192	24.8608	19.4135	15.6398	12.7669
Uniform Prior (unlimited)						
3	7.8501	2.9906	1.3668	0.9785	0.8134	0.6605
5	14.7011	5.1299	2.4118	1.6412	1.3024	1.1594
10	26.6621	11.2580	4.9692	3.4495	2.8239	2.3510
20	35.5904	25.0340	14.1918	7.2049	5.5516	4.7808
40	75.8441	32.2484	25.4009	19.4999	15.9778	12.8562
Maximum-Likelihood						
3	13.5255	4.8866	1.4038	0.9785	0.8261	0.6811
5	77.3067	13.0599	2.6428	1.6595	1.3102	1.1695
10	15393.9516	32.4894	4.9649	3.4777	2.8377	2.3677
20	412.7496	45.5719	12.9859	7.2049	5.5699	4.7808
40	143.0081	37.2404	25.2978	19.7803	16.0145	13.3448

Table 41: Total encoding length (in bits) obtained using the encoding schemes investigated here.

$\sigma \downarrow \ n_p \rightarrow$	7	10	20	40	60	80
$\frac{k}{2}\log n$ (Rissanen)						
3	16.5527	35.0376	75.7733	149.4996	216.4658	288.2756
5	23.4327	42.0958	90.7425	175.6564	261.8794	345.7927
10	29.9255	51.8740	110.1748	216.1499	319.9672	426.7414
20	37.7505	62.5031	129.3881	256.3560	381.7495	506.5641
40	43.2181	71.5888	148.0755	292.6197	436.5068	579.6964
Nohre						
3	65.8804	80.4773	121.0970	198.8118	270.6027	346.8399
5	68.1122	85.0206	132.8533	222.1379	312.6649	401.0218
10	70.1017	90.0173	147.8689	257.9790	366.2997	477.3713
20	71.8971	93.9504	161.6697	293.4050	423.2217	552.5825
40	73.6285	97.4062	174.4685	323.9314	472.8464	620.9369
Uniform Prior (limited)						
3	55.3164	69.1638	108.6340	185.9024	257.5678	333.5565
5	55.4852	72.3204	119.1822	208.1076	298.4225	386.7006
10	55.8602	75.3687	132.8631	242.6101	350.7409	461.6057
20	56.3307	77.7898	145.0341	276.6315	406.2684	535.4532
40	57.5778	80.5509	156.9897	305.9618	454.5542	602.4955
Uniform Prior (unlimited)						
3	64.4443	77.7136	117.7508	195.2983	267.0749	343.3308
5	66.3124	82.3912	129.6408	218.8054	309.3056	397.6425
10	68.4243	87.5196	144.9278	254.8436	363.0899	474.1240
20	69.9752	91.5666	158.8943	290.4033	420.1262	549.4314
40	71.7079	95.0687	171.7836	321.0147	469.8276	617.8553
Maximum-Likelihood						
3	19.4036	42.9776	76.2155	131.2657	174.0642	226.9306
5	33.1719	57.7981	106.3553	183.5838	264.8447	341.9137
10	45.7892	77.0341	145.3034	264.6598	381.0617	503.8287
20	60.7435	96.2573	182.1953	344.8955	504.5480	663.5037
40	69.9605	109.0413	213.9215	412.5046	610.8634	807.9576

Table 42: Polynomial degrees selected by Rissanen's classic ((k/2) log n) method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	0	39	61	0	0	0	0	0	54	17	29	0	0	0	0	86	7	7
5	0	0	0	0	36	64	0	0	0	0	0	41	15	44	0	0	0	0	78	12	10
10	0	0	2	0	41	57	0	0	0	0	0	46	17	37	0	0	0	0	76	17	7
20	0	0	5	6	27	62	0	0	0	8	3	33	21	35	0	0	9	0	69	12	10
40	4	1	8	5	19	63	0	15	2	15	6	24	15	23	21	2	25	7	27	11	7
	np = 40							np = 60							np = 80						
degree ↘ $\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	0	92	5	3	0	0	0	0	95	4	1	0	0	0	0	94	6	0
5	0	0	0	0	96	3	1	0	0	0	0	95	4	1	0	0	0	0	96	3	1
10	0	0	0	0	93	4	3	0	0	0	0	94	5	1	0	0	0	0	97	3	0
20	1	0	0	0	89	6	4	0	0	0	0	97	3	0	0	0	0	0	94	4	2
40	13	1	26	5	49	1	5	9	0	33	0	53	5	0	6	0	19	0	68	6	1

Table 43: Polynomial degrees selected by Nohres method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	2	7	77	14	0	0	0	0	1	98	1	0	0	0	0	0	100	0	0
5	0	0	14	6	66	14	0	0	0	0	2	92	4	2	0	0	0	0	100	0	0
10	16	0	28	3	33	20	0	4	0	13	2	78	3	0	0	0	2	0	95	3	0
20	41	0	28	6	12	13	0	31	0	41	6	20	2	0	10	0	35	0	53	2	0
40	56	2	15	3	6	18	0	70	0	13	5	10	1	1	60	1	18	3	14	4	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0
5	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0
10	0	0	0	0	98	2	0	0	0	0	0	98	2	0	0	0	0	0	99	1	0
20	1	0	5	0	91	2	1	0	0	1	0	98	1	0	0	0	0	0	97	2	1
40	45	1	25	1	26	1	1	23	0	35	0	39	3	0	13	0	21	0	62	4	0

Table 44: Polynomial degrees selected by uniform prior/limited values method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	18	0	59	7	16	0	0	0	0	0	1	99	0	0	0	0	0	0	100	0	0
5	24	0	70	5	1	0	0	0	0	9	4	86	1	0	0	0	0	0	100	0	0
10	44	0	54	2	0	0	0	7	0	43	1	49	0	0	0	0	2	0	96	2	0
20	50	0	48	2	0	0	0	25	0	58	4	12	1	0	10	0	38	0	50	2	0
40	58	4	33	3	2	0	0	57	1	28	5	9	0	0	51	1	26	2	16	4	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0
5	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0
10	0	0	0	0	99	1	0	0	0	0	0	98	2	0	0	0	0	0	99	1	0
20	1	0	6	0	90	2	1	0	0	1	0	99	0	0	0	0	0	0	97	3	0
40	37	1	37	1	23	1	0	19	0	37	0	41	3	0	11	0	25	0	62	2	0

Table 45: Polynomial degrees selected by uniform prior/unlimited values method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	3	7	88	2	0	0	0	0	1	98	1	0	0	0	0	0	100	0	0
5	1	0	15	8	70	6	0	0	0	0	2	92	5	1	0	0	0	0	100	0	0
10	21	0	29	3	33	14	0	4	0	13	2	78	3	0	0	0	2	0	95	3	0
20	50	0	30	5	10	5	0	33	0	40	6	20	1	0	10	0	35	0	53	2	0
40	65	2	16	3	6	8	0	71	0	14	5	9	0	1	60	1	18	3	14	4	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	0	99	1	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0
5	0	0	0	0	100	0	0	0	0	0	0	100	0	0	0	0	0	0	100	0	0
10	0	0	0	0	98	2	0	0	0	0	0	98	2	0	0	0	0	0	99	1	0
20	1	0	5	0	91	2	1	0	0	1	0	98	1	0	0	0	0	0	97	2	1
40	45	1	25	1	26	1	1	23	0	35	0	39	3	0	13	0	21	0	62	4	0

Table 46: Polynomial degrees selected by Maximum-Likelihood-code method.

degree ↘	np = 7							np = 10							np = 20						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	1	44	55	0	0	0	0	0	71	12	17	0	0	0	0	97	3	0
5	0	0	0	2	43	55	0	0	0	0	0	58	19	23	0	0	0	0	94	5	1
10	0	0	6	5	41	48	0	0	0	0	0	65	14	21	0	0	0	0	89	9	2
20	7	0	17	8	22	46	0	10	0	28	5	30	15	12	4	0	32	0	62	2	0
40	26	1	13	4	12	44	0	54	0	16	5	14	3	8	60	1	19	3	13	4	0
degree ↘	np = 40							np = 60							np = 80						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	0	99	1	0	0	0	0	0	97	3	0	0	0	0	0	96	4	0
5	0	0	0	0	98	2	0	0	0	0	0	99	1	0	0	0	0	0	99	1	0
10	0	0	0	0	97	2	1	0	0	0	0	97	3	0	0	0	0	0	97	3	0
20	1	0	5	0	91	2	1	0	0	1	0	97	2	0	0	0	0	0	97	2	1
40	40	1	39	1	18	1	0	21	0	40	0	37	2	0	14	0	27	0	57	2	0

Table 47: Prediction error results for the code-length formulas investigated here. These results are for 7 data points.

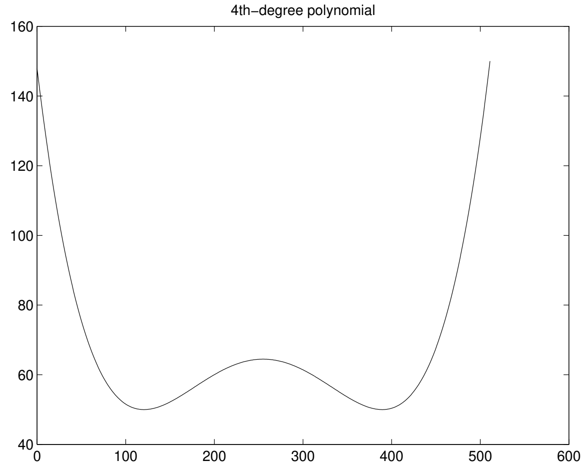
$\sigma\downarrow$ *method* $\rightarrow$	Rissanen ($\frac{k}{2}\log n$)	Nohre	UP-UL	UP-LIM	ML
3	14.8900	8.3955	7.8501	24.2774	13.5255
5	104.4036	64.1869	14.7011	26.8255	77.3067
10	15428.2551	27.1485	26.6621	28.8380	15393.9516
20	442.1098	55.9534	35.5904	32.3318	412.7496
40	188.6660	94.3797	75.8441	36.4271	143.0081

Table 48: Polynomial degrees selected by various methods for the 7-point case.

degree ↘	Rissanen ($\frac{k}{2}\log n$)							Nohre						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	0	39	61	0	0	0	2	7	77	14	0
5	0	0	0	0	36	64	0	0	0	14	6	66	14	0
10	0	0	2	0	41	57	0	16	0	28	3	33	20	0
20	0	0	5	6	27	62	0	41	0	28	6	12	13	0
40	4	1	8	5	19	63	0	56	2	15	3	6	18	0

degree ↘	Uniform Prior (unlim.)							Uniform Prior (limited)							Maximum-Likelihood						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	3	7	88	2	0	18	0	59	7	16	0	0	0	0	0	1	44	55	0
5	1	0	15	8	70	6	0	24	0	70	5	1	0	0	0	0	0	2	43	55	0
10	21	0	29	3	33	14	0	44	0	54	2	0	0	0	0	0	6	5	41	48	0
20	50	0	30	5	10	5	0	50	0	48	2	0	0	0	7	0	17	8	22	46	0
40	65	2	16	3	6	8	0	58	4	33	3	2	0	0	26	1	13	4	12	44	0

D.5.2 Results Covering Only Small Numbers of Points



(a)

Figure 9: Fourth-degree polynomial used in curve fitting experiments.

Experimental parameters:

- Polynomial Coefficients: The results in this subsection are for a fourth-degree polynomial of the form $y = C_4x^4 + C_3x^3 + C_2x^2 + C_1x + C_0$ using the following values.

  $C_4 = 4.437105 \times 10^{-8}$

  $C_3 = -4.525847 \times 10^{-5}$

  $C_2 = 0.01570846$

  $C_1 = -2.125451$

  $C_0 = 147.8588$

  A graph of this function appears in Figure 9.

- (xmin, xmax, ymin, ymax) = [0, 511, 0, 255]
- number of trials at each (σ, n_p) pair = 100
- σ values = 3, 5, 10, 20, 40
- numbers of points (n_p) = 5, 6, 7, 8, 9, 10

Table 49: Prediction error results for the five code-length formulas investigated here.

$\sigma \downarrow \; n_p \rightarrow$	5	6	7	8	9	10
$\frac{k}{2} \log n$ (Rissanen)						
3	47.1464	7.9866	62.7455	107.0238	18.7709	15.1427
5	41.7649	20.7412	56.9978	246.2884	36.6846	7.7209
10	43.7900	35.3650	120.2841	190.8339	45.2975	23.4866
20	62.4317	45.4312	295.6162	577.0922	217.1071	48.9676
40	74.2156	128.7957	579.6920	1383.4135	160.1924	62.9550
Nohre						
3	45.8850	12.3013	9.0609	7.3317	4.3946	2.6210
5	44.6845	24.9187	26.0005	14.9605	5.2593	4.2304
10	42.8168	33.5959	44.6942	36.6494	15.7653	11.8295
20	41.8843	43.4896	125.1446	36.8878	27.1897	24.4531
40	63.0648	65.3228	178.3745	67.2452	42.4717	34.0043
Uniform Prior (limited)						
3	37.5301	30.7577	24.6591	11.7871	4.5953	2.6139
5	36.6976	32.5546	28.9181	15.1915	9.6525	4.7720
10	36.5410	31.9538	30.6239	25.1758	19.9020	15.0794
20	38.6322	34.3966	32.1908	28.7762	25.5678	25.0940
40	46.0359	40.3627	37.1698	34.7799	32.3371	32.5870
Uniform Prior (unlimited)						
3	45.2357	15.0298	9.5792	6.4737	4.0646	2.6210
5	41.6515	30.6159	16.3076	9.2116	5.2593	4.2304
10	42.3007	36.9763	29.1866	26.3522	16.3840	11.8193
20	39.4221	44.8598	64.1025	35.4156	27.4979	24.9010
40	57.9478	52.2569	143.1591	62.2946	42.1998	33.6666
Maximum-Likelihood						
3	47.6891	8.9929	47.6885	31.8277	16.5906	11.7882
5	41.8766	21.1876	30.1372	37.0857	34.4094	5.7525
10	44.7048	35.2528	107.5881	158.3898	39.6507	15.6986
20	62.6929	43.9742	191.6474	508.5603	36.7780	28.9416
40	67.7156	114.4596	533.3916	1020.6461	130.3233	41.4683

Table 50: Total encoding length (in bits; rounded to nearest bit) obtained using the encoding schemes investigated here. Latex generated 24-May-96.

$\sigma \downarrow$ $n_p \rightarrow$	5	6	7	8	9	10
$\frac{k}{2}\log n$ (Rissanen)						
3	24.8103	15.6455	16.1688	19.8573	27.4603	34.3136
5	24.9271	19.8664	23.5811	25.5866	36.0186	42.5834
10	25.7656	24.4448	29.3774	33.8593	44.4580	52.1846
20	27.9629	32.1081	37.4862	40.6102	52.8962	63.0332
40	31.3228	34.2564	41.1045	49.5897	61.8102	72.8342
Nohre						
3	55.1277	60.2356	65.3887	71.3903	75.4258	80.5046
5	55.1184	61.2756	68.1288	73.9723	79.8900	84.8215
10	55.2860	62.1628	69.8910	76.9048	83.7122	90.0602
20	55.9043	63.9348	71.6564	78.8780	86.3574	94.3148
40	57.3161	64.5556	73.5676	81.0331	89.8097	98.4948
Uniform Prior (limited)						
3	41.1854	48.4119	55.3785	60.6481	64.7154	69.1953
5	41.2278	48.5086	55.4150	61.7982	67.2040	72.2222
10	41.1924	48.4906	55.6589	62.3890	69.0131	75.3194
20	41.2889	48.9041	56.2554	63.5643	70.4772	77.9247
40	41.9886	49.3792	57.7901	64.8131	73.1492	81.5417
Uniform Prior (unlimited)						
3	53.7288	59.4522	63.9439	69.2006	72.8078	77.7307
5	53.7130	60.1956	66.3642	71.8251	77.3984	82.1971
10	53.8487	60.8053	68.1663	74.8920	81.3884	87.5681
20	54.3408	62.2580	69.7136	76.8051	84.1155	91.9444
40	55.7769	62.8910	71.8172	78.9054	87.5515	96.1683
Maximum-Likelihood						
3	39.5469	20.3109	18.5826	23.6481	33.4265	42.1057
5	39.6447	28.6248	33.2967	34.8714	50.4147	58.1293
10	41.2155	37.4866	44.9146	50.9125	67.2662	77.4136
20	45.0950	51.9466	60.0468	64.1049	82.4866	97.1424
40	50.9170	54.7435	65.9692	78.8691	97.0289	111.4129

Table 51: Polynomial degrees selected by Rissanen's classic ((k/2) log n) method.

degree ↘	np = 5							np = 6							np = 7						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	27	73	0	0	0	0	0	0	3	97	0	0	0	0	0	0	37	63	0
5	0	0	31	69	0	0	0	0	0	2	4	94	0	0	0	0	0	2	31	67	0
10	0	0	29	71	0	0	0	0	0	2	8	90	0	0	0	0	0	3	26	71	0
20	0	0	32	68	0	0	0	0	0	12	15	73	0	0	0	0	7	7	20	66	0
40	8	1	26	65	0	0	0	7	0	13	13	67	0	0	2	0	12	4	12	70	0
degree ↘	np = 8							np = 9							np = 10						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	0	14	21	65	0	0	0	0	32	21	47	0	0	0	0	39	18	43
5	0	0	0	0	17	15	68	0	0	0	0	36	17	47	0	0	0	0	49	16	35
10	0	0	1	0	21	15	63	0	0	0	1	27	24	48	0	0	0	0	48	21	31
20	0	0	2	2	10	25	61	0	0	6	2	28	18	46	0	0	9	4	41	12	34
40	5	0	8	6	12	7	62	6	0	17	2	21	11	43	14	1	19	9	15	12	30

Table 52: Polynomial degrees selected by Nohre's method.

degree ↘	np = 5							np = 6							np = 7						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	25	0	29	46	0	0	0	1	0	10	10	79	0	0	0	0	3	5	84	8	0
5	32	0	26	42	0	0	0	3	0	15	8	74	0	0	3	0	17	10	56	14	0
10	32	0	28	40	0	0	0	13	0	27	10	50	0	0	12	0	31	13	23	21	0
20	48	0	27	25	0	0	0	35	0	31	6	28	0	0	41	0	30	7	7	15	0
40	54	0	15	31	0	0	0	44	0	20	8	28	0	0	46	0	12	4	9	29	0
degree ↘	np = 8							np = 9							np = 10						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	4	2	82	12	0	0	0	2	1	90	6	1	0	0	0	0	98	2	0
5	0	0	5	4	76	10	5	0	0	1	2	95	2	0	0	0	2	0	94	3	1
10	12	0	34	3	32	6	13	5	0	32	2	53	5	3	4	0	15	3	72	3	3
20	30	0	27	9	15	12	7	33	0	41	0	18	5	3	30	0	45	3	18	2	2
40	62	0	22	4	4	1	7	68	1	13	6	7	3	2	72	0	13	2	9	2	2

Table 53: Polynomial degrees selected by uniform-prior/limited-values method.

degree ↘	np = 5							np = 6							np = 7						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	79	0	21	0	0	0	0	45	0	51	4	0	0	0	20	0	55	9	16	0	0
5	76	0	24	0	0	0	0	52	0	43	5	0	0	0	26	0	61	9	4	0	0
10	77	0	23	0	0	0	0	46	0	52	2	0	0	0	34	0	54	9	3	0	0
20	75	0	25	0	0	0	0	61	0	37	2	0	0	0	48	0	45	6	1	0	0
40	73	1	25	1	0	0	0	66	0	29	5	0	0	0	61	1	35	2	1	0	0
	np = 8							np = 9							np = 10						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	29	5	66	0	0	0	0	5	1	94	0	0	0	0	0	0	99	1	0
5	7	0	40	5	48	0	0	0	0	22	5	73	0	0	0	0	6	0	94	0	0
10	20	0	68	5	7	0	0	8	0	67	1	24	0	0	4	0	44	3	49	0	0
20	38	0	50	8	4	0	0	28	0	62	0	10	0	0	29	0	62	3	6	0	0
40	59	0	35	5	1	0	0	64	2	27	4	3	0	0	58	1	30	3	8	0	0

Table 54: Polynomial degrees selected by uniform-prior/unlimited-values method.

degree ↘	np = 5							np = 6							np = 7						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	39	0	27	34	0	0	0	4	0	15	12	69	0	0	0	0	5	6	86	3	0
5	42	0	29	29	0	0	0	12	0	22	13	53	0	0	5	0	18	11	63	3	0
10	43	0	29	28	0	0	0	24	0	29	10	37	0	0	17	0	30	13	31	9	0
20	58	0	29	13	0	0	0	48	0	30	6	16	0	0	51	0	28	7	6	8	0
40	66	0	14	20	0	0	0	50	0	19	12	19	0	0	56	0	18	3	6	17	0
degree ↘	np = 8							np = 9							np = 10						
$\sigma\downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	4	2	89	5	0	0	0	2	1	94	3	0	0	0	0	0	98	2	0
5	0	0	6	4	79	10	1	0	0	1	2	95	2	0	0	0	2	0	94	3	1
10	16	0	32	3	38	6	5	7	0	33	2	51	5	2	4	0	15	3	73	3	2
20	33	0	28	8	15	10	6	37	0	41	0	16	3	3	32	0	45	3	16	2	2
40	65	0	22	4	4	1	4	74	1	10	6	5	2	2	73	0	13	1	9	2	2

Table 55: Polynomial degrees selected by the Maximum-Likelihood-code method.

degree ↘	np = 5							np = 6							np = 7						
$\sigma \downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	3	0	29	68	0	0	0	0	0	1	6	93	0	0	0	0	0	1	48	51	0
5	4	0	37	59	0	0	0	0	0	3	5	92	0	0	0	0	2	3	47	48	0
10	10	0	27	63	0	0	0	1	0	7	11	81	0	0	0	0	6	4	33	57	0
20	21	0	27	52	0	0	0	4	0	24	14	58	0	0	8	0	17	9	18	48	0
40	33	1	20	46	0	0	0	20	0	19	9	52	0	0	26	0	14	1	11	48	0
degree ↘	np = 8							np = 9							np = 10						
$\sigma \downarrow$	0	1	2	3	4	5	6	0	1	2	3	4	5	6	0	1	2	3	4	5	6
3	0	0	0	1	22	19	58	0	0	0	0	49	17	34	0	0	0	0	65	11	24
5	0	0	0	1	28	19	52	0	0	0	1	50	19	30	0	0	0	0	73	13	14
10	0	0	4	3	37	9	47	0	0	6	1	46	17	30	0	0	3	1	64	19	13
20	6	0	12	4	11	19	48	10	0	24	0	29	13	24	8	0	31	4	39	6	12
40	36	0	8	4	9	5	38	49	0	11	2	12	6	20	62	1	14	3	8	2	10