Polynomial Interpolation for Predicting Decisions and Recovering Missing Data

Oleg Roderick
Mathematics and Computer Science Division
Argonne National Laboratory
oleg@pdx.edu

Ilya Safro
Mathematics and Computer Science Division
Argonne National Laboratory
safro@mcs.anl.gov

ABSTRACT
In this work we improve the existing tools for the recovery and prediction of human decisions based on multiple factors.
We use a latent factor method and obtain the decision-influencing factors from the observed correlations in the statistical information by principal factor identification based on singular value decomposition (SVD).
We generalize on widely used linear representations of decision-making functions by using adaptive high-order polynomial interpolation and applying iterative and adaptive post processing to obtain an estimated probability of every possible outcome of a decision. The novelty of the method consists in the use of flexible, nonlinear predictive functions and in the post processing procedure.
Our experiments show that the approach is competitive with other SVD-based prediction methods and that the precision grows with the increase in the order of the polynomial basis. In particular, the method may be successfully applied when the objective is to get a high number of precisely exact predictions.

Keywords
Prediction, data manifold learning, nonlinear interpolation

1. INTRODUCTION
The ability to predict the outcome of a human, social, or algorithmic decision is important in every customer-oriented field. In many modern applications, such as policy and stock planning, customer recommendation services and social networking, human judgment is now assisted by automatic procedures that analyze large amounts of uniformly structured data about customers and provide fast, repeatable predictions.
In this paper, we propose and demonstrate a method to improve the existing tools for recovery and prediction of decisions based on multiple factors.
We consider the following general situation. A group of decision-makers (*users*) is given access to a set of similar products (*content*). These users evaluate the content by choosing and assigning one of the suggested tags (*ratings*) to some content. Given samples of rated content, we seek to recover, or predict, the missing ratings.
The main contribution of this work is to reinforce the widely used singular-value-decomposition-based principal factor identification by *high-order polynomial interpolation*, instead of using the popular SVD-based *linear* methods. In addition to demonstrated improvement in performance, this reinforcement opens several directions for making the general prediction framework more flexible: for example, by introducing correction variables and combining with the steepest-descent learning procedures.
The method is based on SVD data compression, high-order polynomial interpolation, and subsequent probability estimation. The method is mostly free from assumptions on the structure of the data and is easy to adapt to new situations.
Applied problems of the considered type include ranking scientific papers by predicted preferences of a researcher (Google Scholar-type search [4]), establishing connections between users of a social networking services (LinkedIn-type search [1]), and predicting ratings that customers assign to movies (Netflix problem [8]). The Netflix problem, in particular, provides a large example database; we used small, randomly chosen parts of this database to test the performance of our methods.
In general, for a statistical surface learning problem, the polynomial approximation is often expected outperforms linear approximation – provided there is enough reliable data to construct a high-order surrogate model. For example, high-quality polynomial interpolation was successfully used to predict the behavior of the models exhibiting non-linear dependency with uncertainty on a large number of parameters [5]. It was shown that high-order polynomial approximation produces results of a much better precision than a first-order sensitivity analysis in the quantification of uncertainty for multiphysics models in nuclear engineering. In comparison with linear approximation, the method provides an order of magnitude of increase in quality, for only a modest computational overhead.
In the following sections, we introduce notation and explain our method in comparison with similar approaches; introduce measures of accuracy and prediction, and test the performance of the method.

2. EXPLANATION OF THE METHOD
Approaches to automatic prediction differ from problem to problem. Many successful techniques require additional data, such as advanced classifications of the content, linguistic information, or timing of ratings. Some empirically effective predictions are obtained by *blending* (weighted averaging) results

produced by several prediction methods. The performance is further enhanced by *shrinking*, or detecting and excluding significantly unusual users and seldom-rated content – also, essentially, a weighted averaging applied to the elements of content in the database. On the basic level we find less variety: many approaches amount to a form of linear regression.
We introduce a general, easily implemented unsupervised method that is not limited to linear approximations and does not derive its quality from weighted averaging. We start our discussion with a reference to one basic approach to rating prediction.
Latent factor methods [2] assign a small number of (numerical, unknown) additional characteristics, called *content-factors* to each unit of content; similarly, they assign a small number of *user-factors* to each user. Each unit of content has features that make a particular rating likely, and each user has tendencies to assign particular ratings. The procedure then introduces a *decision-making function* dependent only on content-factors and user-factors, commonly known as *latent factors*. Depending on the definition, the output of the decision-making function may be the predicted rating itself, some real-valued average between several predicted ratings, or some measure of likelihood of a rating that would lead to the choice of the most likely. The assumption is that one can select, from some class of functions, a decision-making function such that the difference between the actual and the predicted ratings is minimal. A latent factor method, then, depends on solving an optimal interpolation problem.
For the purposes of interpolation, the number of relevant variables may be low. However, both the shape of the function and the explicit values of the factors may be unknown and must be interpolated from the known decisions. To do so calls for a simple form of the function.
We point out a special, explicit case of the method where factors are simply all ratings. In this case, the number of relevant factors is very high, but they are all known, allowing for a more flexible function construction.
We present an approach that can be classified as a sequence of two latent factor procedures. The first set of factors is the principal combination of ratings, allowing good compression of the database. The second set of latent factors is created as a measure of deviation of the user from a group of users that agreed on a rating.

2.1 Decision-Making Function

Let us introduce notation. Let $M=\{m_i\}_{i=1}^{N}$ and $U=\{u_i\}_{i=1}^{N}$ be the sets of content and users, respectively. The ratings can be multiple choice – for example, a response o a request to "rate something on the scale from 1 to 5", or binary – for example, 1 stands for "approve a rating r_j", 0 for "do not approve a rating r_j". We denote the multiple choice and the binary discrete assignments of ratings R^d and R^b correspondingly:

$$R^d : U\times M \to \{r_1, r_2, ..., r_K\} \quad (1)$$

$$R^b : U\times M\times\{r_1, r_2, ..., r_K\} \to \{0,1\} \quad (2)$$

where r_i is a possible rating. We introduce a decision-making function θ to approximately estimate the likelihood of each rating:

$$\theta : U\times M \to (-\varepsilon, 1+\varepsilon)\ , \quad (3)$$

That is, $\theta(u_i, m_j, r_k)=\xi$, where $\xi\approx 1$ corresponds to a highly likely rating r_k and $\xi\approx 0$ corresponds to a unlikely rating r_k. We allow a small numerical distortion ε of the interval $(0,1)$ due to interpolation error.
In a limited set of circumstances, direct evaluations of the decision-making function are sufficient for prediction. In general, we find that additional post processing is required in order to distinguish between the values of ξ from the same small neighborhood of 1.
To simplify the discussion, we construct θ as a *user-centered* function. That is, we assume that a decision of a fixed user u_i on an arbitrary unit of content m_j is a function of the decisions of other users on the same unit. Thus, θ is constructed separately for each user and based on all ratings induced by $M_i\subseteq M$ and $F_i\subseteq U$ that are, correspondingly, the subset of content rated by user u_i and the subset of users (including u_i) who rated at least one common content with u_i. An elementary component of θ will be θ_i, written as

$$\begin{aligned}&\theta_i(m_j, r_k; R^d(u_1,m_1), R^d(u_2.m_1), ..., R^d(u_1,m_2), ...) =\\ &\theta_i(m_j, r_k; R^d(u_1,m_1), R^d(u,m) : u\in F_i, m\in M_i)=\xi.\end{aligned} \quad (4)$$

In a reverse *content-centered* approach, a binary version of rating function R^b and various mixes of these types of approaches can be defined similarly. The choice between assumptions and their combinations depends on the dimensions of the database and the structure of sparsity.
Denote by $D^b = D_{u_i}^{b,r_k}$ a user-centered matrix of ratings. Its entry with an index (p,q) is

$$D^b(p,q) = R^b(u_p, m_q, r_k)\ , \quad (5)$$

where $m_q\in M_i$ is a content rated by the user u_i and $u_p\in F_i$. The matrix $D^d = D_{u_i}^{d,r_k}$ is defined similarly. These matrices contain all the available (sparse) multi dimensional information needed for our approach.
To complete the definition, we must decide on the matrix entry corresponding to the unit of content m_q that was not rated by the user u_p. There are several ways to represent this lack of information in D^b and D^d; in general, a representation is chosen to best fit a specific problem. We find that placing 0 in the corresponding entry is appropriate for the matrix D^b, where it represents the simple fact that the rating r_k was not given. The placement of 0 in the matrix D^d implies that an extremely low rating was assigned; placement of any nonzero entry creates significant false information. In our experiments, such entries were filled with random noise (real numbers in the range of possible numerical ratings).
Let $R_{u_i}^{b,r_k}$ be the vector of decisions r_k made by the user u_i:

$$R_{u_i}^{b,r_k} = \begin{pmatrix} R^b(u_i, m_1, r_k) \\ R^b(u_i, m_2, r_k) \\ \vdots \end{pmatrix} \qquad (6)$$

We represent decision-making by a function θ_i constructed in such a way that $\theta_i(m_j, r_k, D_{u_i}^{b,r_k}) - R_{u_i}^{b,r_k}$ is minimal in the L_2 norm. Similarly, we can minimize $\theta_i(m_j, r_k, D_{u_i}^{d,r_k}) - R_{u_i}^{d,r_k}$.

The construction of θ_i is based on high-order polynomial interpolation by collocation, so the number of variables (or principal factors) needs to be low. One way to reduce the list of relevant factors is to compress the information stored in $D_{u_i}^{b,r_k}$.

In our experiments, the compressed form is a projection of the matrix $D_{u_i}^{b,r_k}$ only a lower-dimensional subspace, obtained by SVD decomposition. We define a covariance matrix

$$C = \left(D_{u_i}^{b,r_k}\right) \cdot \left(D_{u_i}^{b,r_k}\right)^T \qquad (7)$$

and choose the basis defined by the matrix $\Phi = (\phi_1, \phi_2, ..., \phi_\eta)$ whose column vectors span the η-dimensional eigenspace of the covariance matrix. Depending on the dimensions of the database, it may be more practical to find the eigenvectors by using decomposition of a smaller matrix $\left(D_{u_i}^{b,r_k}\right)^T \cdot \left(D_{u_i}^{b,r_k}\right)$.

The decision-making function is defined by a multivariable polynomial formula

$$\theta_i = \sum_j x_j \psi_j(S) \ , \qquad (8)$$

where $S = \Phi D^b$ is a compressed form of information and $\Psi = \{\psi_j\}$ is an arbitrary polynomial basis of the redefined interpolating function θ_i. A multivariate basis Ψ is sometimes called a "polynomial chaos" [3]:

$$\psi_j(S) = \psi_j(s_1, s_2, s_3, ...) = \prod_l p^{(k_l)}(s_l) \qquad (9)$$

where s_l is an l th column of S, and $p^{(k_l)}$ is a single variable polynomial of order k_l.

Usually, well-conditioned sets, such as Hermite or Chebyshev polynomials are used as $p(s)$ [6]. However, our best results were achieved with $p^{(0)}(s) = 1, p^{(1)}(s) = s, p^{(2)}(s) = s^2, ...$ leading to the trivial basis.

$$\begin{array}{ll} (0): & 1 \\ (1): & s_1 \quad s_2 \quad s_3 \quad ... \\ (2): & s_1^2 \quad s_1 s_2 \quad s_1 s_3 \quad ... \quad s_2^2 \quad s_2 s_3 \quad ... \\ (3): & s_1^3 \quad s_1 s_2^2 \quad s_1 s_3^2 \quad ... \quad s_1 s_2 s_3 \quad s_1 s_2 s_4 \quad \ldots \end{array} \qquad (10)$$

The coefficients x_j are found by collocation [7]:

$$\sum_j x_j \psi_j(S) = R^b \ . \qquad (11)$$

This system of linear equations needs at least as many rows as there are polynomials in the basis, perhaps slightly more to make up for accidental rank deficiency.

In practice, we fit the size of the basis so that the system includes all available decisions R^b. We then use an optimal least-squares (pseudo-inverse) solution of an overdetermined system.

The number of available entries of R^b limits the maximal order of the polynomial basis and the dimensionality of the reduced information matrix S. For example, for a full basis of order 2 on 30 variables we need at least 496 collocation points, for a full basis of order 3 on 30 variables we need 5456 points. A small number of entries in R^b forces the use of a polynomial basis of small order. Another choice is an incomplete polynomial basis, where only some variables are included in the polynomials of higher degrees, maybe up to a total degree 5. In our experiments, we restrict the dimensionality of S so that there are enough collocation points to use the complete basis sets of orders 2 and 3. In the presence of additional information on the structure and importance of the data, it may be possible to choose the basis adaptively.

Direct application of the function θ to a selection of the content that was already rated and included in D shows an almost perfect recovery of compressed information: for the true ratings r_k, the decision-making function returns values around 1; for the ratings not equal to r_k, the function returns values around 0, with no borderline cases. We observe a dichotomy

$$\begin{array}{l} \sum_k \theta_i(..., r_k) \approx 1 \\ \left|\theta_i(..., r_k) - \theta_i(..., r_{k'})\right| \in \{(-\varepsilon, \varepsilon) \cap (1-\varepsilon, 1+\varepsilon)\} \end{array} \qquad (12)$$

for all k, k'. On the other hand, direct application of θ to unrated content does not always produce meaningful results. In particular, the value $\theta_i(r_k, m_j), m_j \notin M_i$ may fall well outside the range $(0,1)$. For such cases, we conclude that principal factors influencing the rating of m_j were not included in the data compression process.

The following is an essential step of our method. We suggest augmenting matrix D with an additional row corresponding to movie m_j, filled in as if the content was already assigned rating r_k. With an augmented matrix D^{r_k}, we repeat the process of compressing data, constructing the decision-making function and recovering the compressed data. An addition of one row does not significantly alter the decision-making function (in absolute terms, the coefficients x_j do not change much), but it influences the projection Φ: we are now looking at a different cross-section of the same studied surface. Because of the assumption of rating r_k made before data compression, $\theta(..., m_j) \approx 1$.

At this point, our prediction is based on an analysis of an organized collection of outputs of the decision-making function. For each possible rating r_k, we systematically apply the function θ_i to every unit of content actually rated by the user and to the unit of content added in the construction of the augmented form of D. This process produces a set Θ_r of decision-function values

$$\Theta_r = \{\theta_i(m_j, r_1, D^r), \ldots, \theta_i(m_j r_K, D^r)\} \qquad (13)$$

to be post processed for a final prediction.

The only reason for the observed difference between the sets $\Theta_{r_1}, \Theta_{r_2}, \ldots$ is a different assumption about the rating of a single unit of content. Our prediction technique is based on the simple idea that a true assumption will result in a smaller distortion of a high-quality approximation to the true surface. This can be informally justified geometrically (a single, incorrectly included point does not lie on a common smooth surface) or logically (a false assumption spoils the quality of all conclusions).

Experimentally, we will justify two aspects of this statement: (1) that we, in fact, have a high-quality approximation of the true surface, and quality grows with increased order of the polynomial basis; (2) that post processing identifies the correct assumption about the rating, as the one leading to Θ_r that is (in some sense) closest to a version $\Theta_r = \{0,1\}$ with an ideal distinction between predictions.

2.2 Final Rating Prediction

In general, the best way to recover the real value of $R^d(u_i, m_j)$ depends on the particular application for which θ_i is evaluated. In our experiments, sets Θ_r were evaluated, and several different post processing strategies were developed to finalize the discrete prediction. This evaluation is a pairwise independent process and allows us to produce predictions for all needed m_j simultaneously. The estimate for the quality of the prediction can be derived from the same evaluation because the final vector of θ_i results includes the predictions of known $R^d(u_i, m_j)$.

A straightforward post processing strategy consists of choosing a maximum value among all evaluated θ_i predictions for a specific m_j.

It is helpful to reinforce this strategy by a normalization process over all known predicted ratings. In particular, θ_i can be turned into a probabilistic function by scaling all prediction results to a binary vector. This strategy has produced the best results in situations when only one dominating $R^d(u_i, m_j)$ result was observed among the K possible.

A more powerful method is a weighted strategy in which a final rating is calculated by

$$\sum_{k=1}^{K} r_k \alpha_k \Big/ \sum_{k=1}^{K} \alpha_k \, , \qquad (14)$$

where α_k is a weighting coefficient that can be calculated and learned in several ways. In our experiments, good results were achieved for

$$\alpha_k = \theta_i(m_j, r_k, D_{u_i}^{b, r_k}) \, , \qquad (15)$$

and slightly better results achieved by a version corrected by the size of the group of users that agreed on a rating:

$$\alpha_k = \theta_i(m_j, r_k, D_{u_i}^{b, r_k}) \cdot \left| F_{i, m_j, r_k} \right| \qquad (16)$$

with $F_{i, m_j, r_k} = \{u \in F_i : R^d(u, m_j) = r_k\}$.

Weighted final prediction can be reinforced by introducing various correction variables for θ_i and F_{i, m_j, r_k}. One approach is a simple, fixed-point iteration technique, with a small number of identical steps. At each step, we introduce a correction term Δ and modify (15) to a form

$$\alpha_k = \theta_i(m_j, r_k, D_{u_i}^{b, r_k}) \cdot (1 + \Delta) \, . \qquad (17)$$

We have also attempted a polynomial structure for the correction term:

$$\begin{aligned} \theta_i &= \theta_i(m_j, r_k, D_{u_i}^{b, r_k}) \\ \alpha_k &= \theta_i \cdot (1 + \Delta_0 + \theta_i \Delta_1 + \theta_i^2 \Delta_2 + \ldots). \end{aligned} \qquad (18)$$

We then obtained the optimal values of Δ by interpolation, using the content that received rating as training points, updated the decision-making functions by $\theta_i := \theta_i \cdot (1 + \Delta)$, and repeated the step. In many cases, this approach resulted in a small improvement in the quality of post processed results. However, the optimal post processing technique remains to be found.

3. NUMERICAL RESULTS

The numerical experiments were performed on randomly chosen data points (u_i, m_j, r_k) from a Netflix contest database [3]. During each experiment, 10,000 points were extracted and their ratings predicted. Other information (provided by Netflix) related to the chosen triples was ignored.

A rooted mean-square error was suggested by the contest as a measure of prediction success, and defined as

$$RMSE = \sqrt{\frac{1}{n} \sum (r_{true} - r_{predicted})^2} \, . \qquad (19)$$

To get an upper bound for the prediction, we first measured the RMSE obtained using simple averaging of the ratings given by all users. This approach gave a RMSE upper bound of 1.12 .

A simple linear method (the interpolating basis consisted of only linear functions, an approach that is approximately equivalent to generic SVD-based techniques used by many of the Netflix contest participants) improved the averaging method by approximately 10%. Since this method highly depends on the number of latent factors, we experimented with $5 < \eta < 30$. The observed variability of RMSE was approximately 0.5%.

The next series of experiments was performed using higher interpolation orders, namely 2 and 3. We then observed an improvement of final RMSE by 0.05 and 0.1, respectively.

For our specific problem, using even higher orders of interpolation severely restricted the number of factors. Also, it often caused numerical instability for several data points (thus influencing the whole post-processing procedure), which prevents us from presenting the corresponding final results. In principle, such numerical instability can be eliminated by various strategies (including preconditioned collocation and the sensitivity analysis with identification and elimination of sources of instability) that are beyond the scope of current work.

The average results of the experiments are presented in Table 1.

Table 1. Results

Algorithm	**Order**	**Average RMSE**
Averaging	-	1.12
Linear interpolation	1	0.98
High-order polynomial interpolation	2	0.95
High-order polynomial interpolation	3	0.90

Clearly, our current numerical results are worse than those obtained by the best contest participants. As a basic information matrix (for SVD and polynomial/linear latent factor analysis) we used only a user-centered matrix that is significantly less informative than a full matrix. On the other hand, using such a small base information matrix reduced the running time of our experiments significantly. Our main goal is to present a new way of working with latent factors and to demonstrate that high-order polynomial interpolation can successfully extend a popular linear approach.

4. CONCLUSIONS

We presented a generalization of a widely used linear method for working with the latent factors of an information model. The generalization consists of a high-order polynomial interpolation scheme rather than linear combination. The approach is highly adaptive and can be reinforced by iterative parameter learning methods.

It is remarkable that an approach originally developed for uncertainty quantification of physical systems with no conscious actors has also produced a successful recovery of human decisions.

The experiments on data with limited information content shows a significant improvement in comparison to the linear method. The improvement increased with the change of the interpolation order from 2 to 3.

Overall, the method appears to be competitive in its class, has a moderate implementation and computational cost when applied to new data sets, and can be combined with sophisticated post processing techniques. We recommend considering the high-order polynomial interpolation scheme for data recovery and prediction algorithms that are based on latent factors extraction.

Acknowledgments

This work was supported in part by the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract DE-AC02-06CH11357. We express our gratitude to Dr. Mihai Anitescu for his advice.

REFERENCES

[1] Adamic L. and Adar E. 2005. How to search a social network. Social Networks, 27, 3, 187-203.

[2] Aragwal D. and Merugu S. 2007. Predictive discrete latent factor models for large scale dyadic data, proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining, 26-35.

[3] Ghanem R. and Spanos P. 1990. Polynomial chaos in stochastic finite elements. Journal of Applied Mechanics, 57, 1, 1999, 197-203.

[4] Jasco P., 2005. Google Scholar: The pros and the cons. Online Information Review, 29, 2, 208-214.

[5] Roderick O., Anitescu M., and Fischer P. 2008. Stochastic finite element approaches using derivative information for uncertainty quantification. Nuclear Science and Engineering, preprint ANL/MCS 1551-1008.

[6] Lorentz R. 2000. Multivariate Hermite interpolation by algebraic polynomials: A survey. Journal of Computational and Applied Mathematics, 122, 2000, 167-201

[7] Xiu D. and Hesthaven J. 2005. High-order collocation methods for differential equations with random inputs. SIAM Journal on Scientific Computing, 27, 2005, 1118-1139.

[8] Netflix, 2006. The Netflix Prize rules. http://www.netflixprize.com/rules