FAST LEAST-SQUARES POLYNOMIAL APPROXIMATION IN MOVING TIME WINDOWS

Erich Fuchs, Klaus Donner

Faculty for Mathematics and Computer Science, University of Passau
Innstr.33, D-94032 Passau, Germany
email: (fuchs|donner)@fmi.uni-passau.de

ABSTRACT

Only a few time series methods are applicable to signal trend analysis under real-time conditions. The use of orthogonal polynomials for least-squares approximations on discrete data turned out to be very efficient for providing estimators in the time domain. A polynomial extrapolation considering signal trends in a certain time window is obtainable even for high sampling rates. The presented method can be used as a prediction algorithm, e.g. in threshold monitoring systems, or as a trend correction possibility preparing the analysis of the remaining signal. In the theoretical derivation, the recursive computation of orthogonal polynomials allows the development of these fast algorithms for least-squares approximations in moving time windows.

1. INTRODUCTION

It is well-known that the orthogonal sequence of *Discrete Chebyshev Polynomials* can be used for equally weighted least-squares approximation on discrete data sampled with uniform separation [1, 2]. With the definitions ($\mathcal{P}_n(\mathbb{R},\mathbb{R})$ is the vectorspace of polynomials with degree less or equal n)

$$\begin{aligned} \Delta, T &: \mathcal{P}_n(\mathbb{R},\mathbb{R}) \to \mathcal{P}_n(\mathbb{R},\mathbb{R}), \\ \Delta f(x) &:= f(x+1) - f(x) \text{ (difference operator)}, \\ Tf(x) &:= f(x+1) \text{ (shift operator) and} \\ q &: \mathbb{R} \to \mathbb{R}, \quad q(x) := x(x-m-1), \end{aligned}$$

the n-th Discrete Chebyshev Polynomial q_n (with leading coefficient 1) can be represented as

$$\begin{aligned} q_n(x) &= \frac{n!}{(2n)!}\,\Delta^n \left(\prod_{j=0}^{n-1} T^{-j} q(x) \right) \\ &= \frac{n!}{(2n)!}\,\Delta^n \left(\frac{\Gamma(x+1)\,\Gamma(x-m)}{\Gamma(x+1-n)\,\Gamma(x-m-n)} \right) \end{aligned}$$

for $n \le m$, where Γ is the gamma function and $m+1$ the number of samples used for the approximation [1, 3].

These orthogonal polynomials fulfill the three-term recurrence relation

$$q_{i+1}(x) = \left(x - \frac{m}{2}\right) q_i(x) - \delta_i\, q_{i-1}(x)$$

for $i = 1, \ldots, n-1$, where $q_0(x) = 1$, $q_1(x) = x - \frac{m}{2}$ and $\delta_i := \dfrac{i^2\,((m+1)^2 - i^2)}{4\,(4\,i^2 - 1)}$ with $i \ge 1$ [3].

With $S_m : \mathcal{P}_n(\mathbb{R},\mathbb{R}) \to \mathbb{R}$, $S_m(p) := \sum_{j=0}^{m} p(j)$, an inner product $\langle \cdot \,|\, \cdot \rangle$ for the vectorspace $\mathcal{P}_n(\mathbb{R},\mathbb{R})$ is defined by

$$\langle p \,|\, q \rangle := S_m(pq) = \sum_{j=0}^{m} p(j) q(j).$$

The least-squares approximation with a polynomial of degree n for the data vector $(y_0, y_1, \ldots, y_m)$ is then given by

$$p = \sum_{i=0}^{n} \frac{\sum_{j=0}^{m} y_j\, q_i(j)}{S_m(q_i^2)}\, q_i.$$

In real-time applications, which process a stream of continuously arriving data, it is often useful that a polynomial p approximating the last $m+1$ values is computed after each receipt of a new value. This means that a processing in moving time windows is required. The orthogonal polynomials q_i and the norming factors $1/S_m(q_i^2)$ are independent of the data to be approximated; therefore they can be calculated once and for all, provided that m and n remain fixed. The straight forward computation of $\sum_{j=0}^{m} y_j\, q_i(j)$, however, is time consuming, especially if m is large, which is the case in several applications.

More specific, let $y_0, \ldots, y_t, y_{t+1}, \ldots, y_{t+m}, y_{t+m+1}, \ldots$ be the measuring data; then the polynomials approximating the data $(y_t, \ldots, y_{t+m})$ and $(y_{t+1}, \ldots, y_{t+m+1})$ respectively are given by

$$p_t(x) = \sum_{i=0}^{n} \frac{\alpha_{i,t}}{S_m(q_i^2)}\, q_i(x) \text{ and}$$

$$p_{t+1}(x) = \sum_{i=0}^{n} \frac{\alpha_{i,t+1}}{S_m(q_i^2)} q_i(x)$$

with $\alpha_{i,s} := \sum_{j=0}^{m} y_{s+j}\, q_i(j)$, $i = 0, \ldots, n$, provided that the origin of the coordinate system moves together with the considered time window.

It is the purpose of this paper to develop a quick update algorithm from $\alpha_{i,t}$ to $\alpha_{i,t+1}$ independent of the approximation length m. This allows the implementation of a moving least-squares approximation for real-time applications using polynomial estimators by analogy with a moving average.

2. LEAST-SQUARES APPROXIMATION IN MOVING TIME WINDOWS

Since $(q_0, \ldots, q_i)$ is a basis of $\mathcal{P}_i(\mathbb{R}, \mathbb{R})$, Tq_i can be written as $Tq_i = \sum_{l=0}^{i} \gamma_l^{(i)} q_l$ with some $\gamma_l^{(i)} \in \mathbb{R}$. Since q_i and Tq_i have leading coefficient 1, this equation leads to $T^{-1}q_i - q_i = -\sum_{l=0}^{i-1} \gamma_l^{(i)} T^{-1} q_l$. Given these coefficients, it is possible to prove the following assertion:

Theorem 1: For $i = 0, \ldots, n$ the values $\alpha_{i,t+1}$ can be represented as

$$\alpha_{i,t+1} = \alpha_{i,t} + y_{t+m+1}\, q_i(m+1) - y_t\, q_i(0) - \sum_{l=0}^{i-1} \gamma_l^{(i)} \alpha_{l,t+1}$$

with $\sum_{l=0}^{-1} \gamma_l^{(0)} \alpha_{l,t+1} := 0$ and $\gamma_l^{(i)}$ as above.

Proof: For $i = 0, \ldots, n$ the following is valid:

$$\begin{aligned}
\alpha_{i,t+1} &= \sum_{j=0}^{m} y_{t+1+j}\, q_i(j) \\
&= \sum_{j=1}^{m+1} y_{t+j}\, q_i(j) + \sum_{j=1}^{m+1} y_{t+j} \left(T^{-1}q_i(j) - q_i(j)\right) \\
&= \alpha_{i,t} + y_{t+m+1}\, q_i(m+1) - y_t\, q_i(0) + \\
&\qquad \sum_{j=1}^{m+1} y_{t+j} \left(-\sum_{l=0}^{i-1} \gamma_l^{(i)} T^{-1} q_l(j) \right) \\
&= \alpha_{i,t} + y_{t+m+1}\, q_i(m+1) - y_t\, q_i(0) - \sum_{l=0}^{i-1} \gamma_l^{(i)} \alpha_{l,t+1}.
\end{aligned}$$

With these formulas it is possible to compute $\alpha_{i,t+1}$ iteratively starting from $i = 0$ with a fixed number of arithmetic operations independent of m.

It remains to determine the coefficients $\gamma_l^{(i)}$.

Theorem 2: Let $q_{-1} := 0$, $\gamma_{-1}^{(i)} := 0$ and $\gamma_j^{(i)} := 0$ for $j > i \geq -1$; then $\gamma_0^{(0)} = 1$ and for $i = 0, \ldots, n-1$ and $l = 0, \ldots, i+1$ the recursive relation

$$\gamma_l^{(i+1)} = \gamma_{l-1}^{(i)} + \delta_{l+1} \gamma_{l+1}^{(i)} + \gamma_l^{(i)} - \delta_i \gamma_l^{(i-1)}$$

holds with δ_i defined as above.

Proof: $\gamma_0^{(0)} = 1$ follows from $q_0 = 1 = Tq_0$. Multiple use of the recurrence relation for the orthogonal polynomials allows the transformations

$$\begin{aligned}
&\sum_{l=0}^{i+1} \gamma_l^{(i+1)} q_l = Tq_{i+1} = T(x - \frac{m}{2}) Tq_i - \delta_i Tq_{i-1} \\
&= (x + 1 - \frac{m}{2}) \sum_{l=0}^{i} \gamma_l^{(i)} q_l - \delta_i \sum_{l=0}^{i-1} \gamma_l^{(i-1)} q_l \\
&= \sum_{l=0}^{i} \gamma_l^{(i)} (x - \frac{m}{2}) q_l + \sum_{l=0}^{i} \gamma_l^{(i)} q_l - \delta_i \sum_{l=0}^{i-1} \gamma_l^{(i-1)} q_l \\
&= \sum_{l=0}^{i} \gamma_l^{(i)} (q_{l+1} + \delta_l q_{l-1}) + \sum_{l=0}^{i} \gamma_l^{(i)} q_l - \delta_i \sum_{l=0}^{i-1} \gamma_l^{(i-1)} q_l \\
&= \sum_{l=1}^{i+1} \gamma_{l-1}^{(i)} q_l + \sum_{l=-1}^{i-1} \gamma_{l+1}^{(i)} \delta_{l+1} q_l + \\
&\qquad \sum_{l=0}^{i} \gamma_l^{(i)} q_l - \delta_i \sum_{l=0}^{i-1} \gamma_l^{(i-1)} q_l \\
&= \gamma_i^{(i)} q_{i+1} + \gamma_{i-1}^{(i)} q_i + \gamma_i^{(i)} q_i + \\
&\qquad \sum_{l=0}^{i-1} (\gamma_{l-1}^{(i)} + \delta_{l+1} \gamma_{l+1}^{(i)} + \gamma_l^{(i)} - \delta_i \gamma_l^{(i-1)}) q_l.
\end{aligned}$$

The assertion follows by comparison of the respective coefficients.

For polynomials with small degree, it is possible to derive formulas for the direct computation of $\gamma_l^{(i)}$.

Corollary 1: The following equations for $\gamma_{i-s}^{(i)}$ hold provided that $i \geq s$ for each s:

$$\gamma_{i-k}^{(i)} = \binom{i}{k} \text{ for } 0 \leq k \leq 2,$$

$$\gamma_{i-3}^{(i)} = \binom{i}{3} + 2 \sum_{j=1}^{i-2} \delta_j - (i-2)\delta_{i-1},$$

$$\gamma_{i-4}^{(i)} = \binom{i}{4} + (2i-3) \sum_{j=1}^{i-3} \delta_j - \binom{i-2}{2} (\delta_{i-2} + \delta_{i-1}),$$

$$\gamma_{i-5}^{(i)} = \binom{i}{5} + (i-2)^2 \sum_{j=1}^{i-4} \delta_j - \binom{i-2}{3} (\delta_{i-3} + \delta_{i-2} +$$

$$\delta_{i-1}) + 2 \sum_{j=1}^{i-4} \delta_j \left(\sum_{k=1}^{j+1} \delta_k - \sum_{k=j+2}^{i-1} \delta_k \right) + (i-4)\delta_{i-3}\delta_{i-1}.$$

This can be proven by induction using the recursive computation formula of theorem 2.

Altogether these results provide a possibility to compute polynomial approximations efficiently for given degree n (usually ≤ 3 in practice) and approximation length m. For processing in moving time windows with small $m (\leq n+2)$ a direct computation of $\alpha_{i,t}$ taking advantage of the symmetry of q_i is preferable. In other cases, if m is large, costs of $(n+1)(n+4)/2$ additions and less than $n(n+5)/2$ multiplications result from theorem 1 for one update step from p_t to p_{t+1} assuming that the data independent values $q_i(0)$, $q_i(m+1)$ and $\gamma_l^{(i)}$ were computed in advance.

$$
\begin{aligned}
\alpha_{0,t+1} &= \alpha_{0,t} + y_{t+m+1} - y_t \\
\alpha_{1,t+1} &= \alpha_{1,t} + (1+\frac{m}{2})y_{t+m+1} + \frac{m}{2}y_t - \alpha_{0,t+1} \\
\alpha_{2,t+1} &= \alpha_{2,t} + \frac{m^2+5m+6}{6}y_{t+m+1} - \frac{m^2-m}{6}y_t \\
&\quad -2\alpha_{1,t+1} - \alpha_{0,t+1} \\
\alpha_{3,t+1} &= \alpha_{3,t} + \frac{m^3+9m^2+26m+24}{20}y_{t+m+1} + \\
&\quad \frac{m^3-3m^2+2m}{20}y_t - \\
&\quad 3\alpha_{2,t+1} - 3\alpha_{1,t+1} - \frac{m^2+2m+12}{10}\alpha_{0,t+1}
\end{aligned}
$$

Table 1: Explicit update formulas

Therefore the number of arithmetic operations required for one update step depends on the degree of the approximating polynomial only. This means that the developed algorithm is especially suitable for approximations in large time windows. Table 1 shows the explicit computation of the first few $\alpha_{i,t+1}$.

If the approximation length m may change during a real-time application, the computation of $q_i(m+1)$ and $q_i(0)$ in advance is impossible. The following simplification avoids the evaluation of the orthogonal polynomials during the update step: since the coefficients $\beta_{i,j}$ of $q_i(x) = \sum_{j=0}^{i} \beta_{i,j} x^j$ are known, $q_i(0) = \beta_{i,0}$ and $q_i(m+1) = (-1)^i q_i(-1) = (-1)^i \sum_{j=0}^{i} (-1)^j \beta_{i,j}$ holds due to symmetry reasons.

3. APPLICATIONS

In general, the new method may be used to figure out threshold crossings in real-time considering polynomial trends in the signal. The least-squares approximation smoothes the signal and the essential behaviour of the signal (ascent or descent and their intensity) is contained in the approximating polynomial. By evaluating this polynomial a short period in the future (extrapolation), it is possible to predict the signal behaviour. Furthermore a polynomial trend correction for a signal is possible as a signal preprocessing step with each receipt of a new value. This avoids e.g. misinterpretations of low frequencies due to long-term signal trends in a subsequent fourier analysis of the remaining signal.

3.1. Tool monitoring systems

A typical application for checking a threshold crossing is a very important and time critical aspect of monitoring a CNC-lathe, namely the collision detection based on measured force signals. For this purpose we have implemented the described method on a Digital Signal Processor TMS320C31 (Texas Instuments, 32-bit floating-point arithmetic). Using a polynomial of degree 2 approximating the last 64 values at a time we have shown that the reaction time for detecting a threshold crossing in the case of a collision could be significantly reduced compared to moving average algorithms; this could have been achieved without a worsening of the false alarm behaviour. Besides the original cutting force signal figure 1 shows the continuously computed extrapolation values of the moving polynomial approximation together with two different moving averages.

Besides the collision detection, the moment of the first contact between the cutting tool and the workpiece may also be controlled by the described method: the value $\alpha_{1,t}$, which represents mostly the ascent behaviour of a signal, is a reasonable indicator for the moment of the first contact.

3.2. General implementation aspects

Several aspects had to be considered for a thorough and efficient implementation. The parameters $\alpha_{i,t}$, which can be computed *in place* due to the formula in theorem 1, were set to 0 initially thus avoiding a time consuming computation for the first approximating polynomial. The high sampling rates (e.g. up to 10 kHz in the monitoring systems) together with long monitoring intervals require a careful analysis of possible rounding errors due to the iterative computation of $\alpha_{i,t}$. Rounding problems may already arise while computing the moving average $\alpha_{0,t}$ due to the addition of the relatively small y_{t+k} to the previous value of $\alpha_{0,t}$. Thus an increasing rounding error may be propagated to the remaining $\alpha_{i,t}$ with $i > 0$ causing additional trouble. These numerical problems can be avoided on the DSP by using either a 40-bit addition algorithm (which needs about 70 (50) processor cycles for the update step

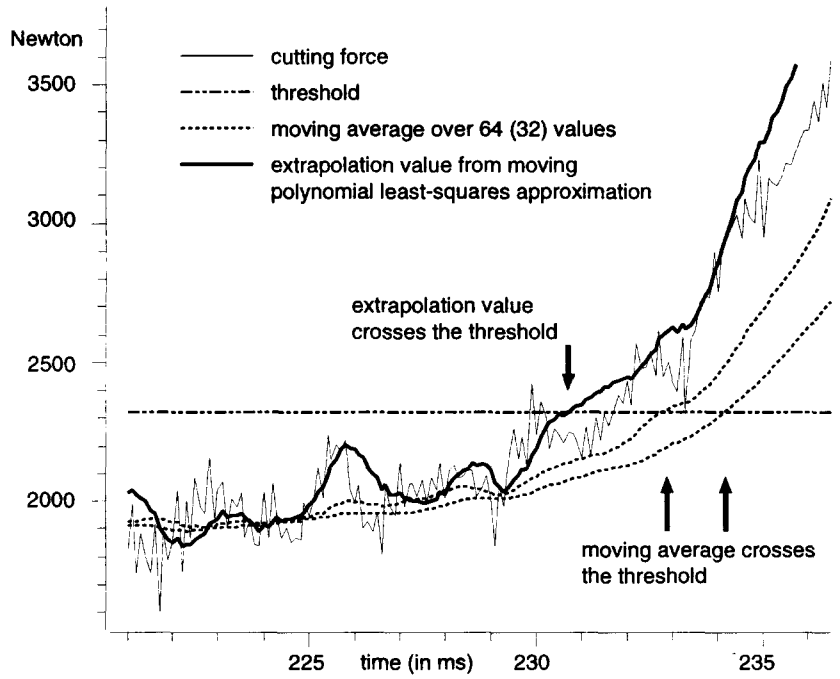

Figure 1: Threshold crossing detection

from $\alpha_{i,t}$ to $\alpha_{i,t+1}$ for a polynomial of degree 3 (2)) or by implementing an analogue of the Kahan-Babuška-summation [4]. This summation method is generally suitable for the floating-point addition of small numbers to already big sums. Table 2 shows as an example pseudo code for an improved computation of $\alpha_{0,t}$.

```
S := 0; δ_s := 0;
LOOP over t {
    x     :=  y_{t+m+1} − y_t;
S_new     :=  S + x;      /*possible rounding error*/
    x̃     :=  S_new − S;  /*compute correction*/
    δ_x   :=  x − x̃;
    δ_s   :=  δ_s + δ_x;  /*add correction*/
    S     :=  S_new;
  α_{0,t} :=  S + δ_s;    /*corrected sum*/
}
```

Table 2: Kahan-Babuška-summation

The evaluation of the polynomial $p_t(x)$ for a certain x can be done very fast due to the usage of the Clenshaw-algorithm, which is generally applicable to orthogonal polynomials fulfilling a three-term recurrence relation.

3.3. Object tracing

Tracing objects in image sequences is another field of application of the presented method. By considering each coordinate separately, it is possible to estimate accurately the centre (or other significant points) of an object in a subsequent image by extrapolating a polynomial of degree 2 approximating the centres in the previous images; this allows the determination of a region of interest, in which the traced object has to be searched for.

A two-dimensional version of the moving polynomial approximation has been developed for the preprocessing of images. The usage of a tensor product approach for two dimensions allows fast algorithms (e.g. for gradient calculation) providing results for further evaluation and interpretation (e.g. edge detection, object matching) in image sequences.

4. REFERENCES

[1] T. S. Chihara; *An Introduction to Orthogonal Polynomials*; Gordon and Breach, Science Publishers, New York, 1978

[2] F. B. Hildebrand; *Introduction to Numerical Analysis*; Dover Publications, New York, 1987

[3] K. Donner; *Orthogonale Polynome für zeitäquidistante Abtastung von Signalen*; MIP-Bericht 9515 der Fakultät für Mathematik und Informatik der Universität Passau, 1995

[4] G. Maess; *Vorlesungen über numerische Mathematik I*; Birkhäuser Verlag, Basel, 1985