Predicting Physical Processes in the Presence of Faulty Sensor Readings

Matthew Clegg and Keith Marzullo*

Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093

Abstract

A common problem in the operation of mission critical control systems is that of determining the future value of a physical quantity based upon past measurements of it or of related quantities. Some of the sources of variability that make this problem difficult include imprecision due to measurement error, measurements that change with time, and intermittent failures leading to faulty measurements.

We present a novel solution to this problem based upon the following hypotheses: (1) Each measurement is presented as a timestamped interval that contains the correct value of the physical quantity; (2) The underlying process describing the physical quantity is linear or a low-degree polynomial; and, (3) up to f of the n measurements may be arbitrarily faulty, where $n \geq 2(f+1)$. For a given future time t_0, our algorithm produces the smallest set of possible values that the function may take at time t_0. For linear functions, our algorithm runs in time $O(n^2)$, and for degree-d polynomials it runs in time $O(n^{d+1})$.

1 Introduction

Consider the following problem that arises in mission-critical systems. A program measures a time-varying physical value $y(t)$ through a sensor in order to determine whether an action should be taken. Due to the mission-critical nature of the system, failure to take the action when necessary is catastrophic. However, taking the action unnecessarily is expensive, and so should be avoided.

We model the need to take the action by a condition $P(y)$, which when true indicates that the action should be taken. We say that the program has computed a *false positive* if it computes $P(y)$ is true when there is no need to take the action, and it has computed a *false negative* if it computes $P(y)$ is false when the action must be taken. We wish to eliminate all false negatives and minimize false positives.

Three problems make computing $P(y)$ difficult. First, the physical value y varies with time. Second, the sensor measurements are not precise: there is some measurement error with each reading. Third, some of the measurements may be incorrect.

A traditional approach would be to assume that the physical value y can be described by a function that is linear in t over a short period of time. One then computes the least-squares fit to the measurements, and then computes the expected value of y at the time of interest. Unfortunately, this method can result in both false positives and in false negatives when even only one measurement is incorrect. Another similar approach is to fit a function to the measurements using some knowledge of the distribution of the measurement errors of y and using an expectation minimization algorithm [DLR]. However, this too can result in both false positives and in false negatives even if only one measurement is incorrect.

A nonparametric method of dealing with imprecision of measurement is to represent each measurement as an interval. If the measurement is correct, then the physical value is contained in this interval. The program asserts $P(y)$ to be true if any point in the interval satisfies P. If the measurement is correct, then false negatives are avoided. In order to minimize false positives, the interval should be as small as possible.

If the interval is incorrect, however, then false negatives may still occur. If a number n of measurements of y can be made simultaneously, then we can use the fault-tolerant intersection algorithm described in [M] to compute the smallest correct interval under the assumption that no more than f of the n measurements are incorrect. This algorithm is optimal in the sense that it computes the smallest interval that has the property that any value that is supported by at least $n-f$ of the readings will be contained in it. This algorithm runs in time $O(n \log n)$.

If the measurements are not made simultaneously, we can adapt the above technique by assuming that that the rate of the change of $y(t)$ is bounded: there exist constants $(dy/dt)_{min}$ and $(dy/dt)_{max}$ such that

$$(dy/dt)_{min} \leq dy/dt \leq (dy/dt)_{max}.$$

Given a measurement $u \leq y(t_1) \leq v$, we can scale this measurement to a time t_0 using the formula

$$u+(t_0-t_1)(dy/dt)_{min} \leq y(t_0) \leq v+(t_0-t_1)(dy/dt)_{max}.$$

*This research was supported by AFOSR Grant F49620-96-1-0215.

Having scaled all measurements to a common time t_0, we can use the method described above.

This method suffers from three problems. First, it assumes that we can find the constants $(dy/dt)_{min}$ and $(dy/dt)_{max}$, which may not be the case. Second, even when they can be found, one is often forced to use pessimistic values that make the scaled readings very imprecise, especially for old measurements. Finally, this method of scaling is pessimistic if one can assume that y can be described by a polynomial of t.

In this paper, we trade the assumption of a bounded rate of change of y with the assumption that $y(t)$ is linear or a low-degree polynomial. Doing so allows us to compute a much smaller interval. Furthermore, our method is useful even when some measurements are old. Our method is self-calibrating, in that it implicitly computes bounds on the rate of change of y.

This problem first came to our attention through our work in fault-tolerant clock synchronization [CM]. It is common to assume that clocks drift apart at a bounded rate, but we have found empirically that clocks drift apart at a constant rate over a long period of time. For example, in our laboratory we have a set of commercial PCs. In this environment, we found that the relative drift rates of the clocks remain constant over weeks at a time.

We were led to ask whether or not this property of clock rates could be used to achieve tighter clock synchronization. Consider a processor i that makes n measurements of the value of a remote processor j's clock over a short period of time, such as a minute. We assume that no more than f of the n measurements can be incorrect. If we also assume that i and j's clocks drift apart at a constant rate over this period, then it follows that there is a linear relationship between the values of i's clock and j's clock. Under these assumptions, we ask how precisely can i estimate the value of j's clock at a specific future point in time?

In this paper, we answer a more general question. We consider a sensor that measures a physical value that over an interval of time of interest can be described as an unknown low-degree polynomial function. Given n measurements of the physical value where no more than f of the measurements are incorrect and a future time t_0, we compute an interval that contains all possible values of the physical value at time t_0. Moreover, our algorithm is optimal in the sense that it computes the smallest such interval.

The remainder of this paper is organized as follows. We first give an explicit definition of the problem that we solve. Next, we show how this problem can be solved when the measured physical value is linear with respect to time. Finally, we show how our results can be extended to degree-d polynomials.

2 Problem Statement

We first model the values returned by sensors.

Definition 1 (sensor reading) *Sensor readings are of the form $(t, [u, v])$, meaning that at time t, the value of the physical variable was known to be in the interval $[u, v]$.*

We then define what it means for a function that describes a physical value to be supported by a set of sensor readings.

Definition 2 (support) *Let $S = \{r_1, r_2, ..., r_n\}$ be a set of n sensor readings, where each $r_i = (t_i, [u_i, v_i])$. We say that a function $y(t)$ is* supported *by S if $y(t_i) \in [u_i, v_i]$ for each i. The* support *of S is the set of all functions that are supported by S.*

Definition 3 (f-resilient support) *Let $S = \{r_1, r_2, ..., r_n\}$ be a set of sensor readings. We say that that a function $y(t)$ has f-*resilient support *by S if there is a subset S' of S such that $|S'| = n - f$ and y is supported by S'. The f-*resilient support *of S is the set of all functions y such that y has f-resilient support by S.*

Definition 4 (f-resilient envelope) *We say that a point (t, u) is in the f-*resilient envelope *of S if there is a function $y(t)$ that has f-resilient support by S and where $y(t) = u$.*

We can now define the problem considered in this paper.

Definition 5 (fault-tolerant sensor prediction) *Given a collection S of readings and a distinguished time t_0, find the smallest interval $[u, v]$ such that every point (t, w) which is in the f-resilient envelope of S satisfies $u \le w \le v$.*

3 The Linear Case

In this section, we consider the linear case: given a collection of readings $(t_1, [u_1, v_1]), \ldots, (t_n, [u_n, v_n])$, and a distinguished time t_0, we show how to compute the smallest interval $[u, v]$ that satisfies the following property:

linear prediction If $y = at + b$ is a function which is supported by at least $n - f$ of the readings $(t_i, [u_i, v_i])$, then $y(t_0) \in [u, v]$.

Given a collection S of readings, we define the f-*resilient linear support* of S to be the set of linear functions that are in the f-resilient support of S. Similarly, we define the f-*resilient linear envelope* of S to be the set of points (t, u) such that there exists a linear function y that has f-resilient support in S and that satisfies $y(t) = u$.

As discussed in the introduction, this framework models the divergence between clocks under the assumption of constant drift rate.

3.1 Example

In order to motivate our algorithm, we give an example. Figure 1 depicts two readings. The first reading r_1 is $(1, [1, 3])$, and the second r_2 is $(2, [2, 4])$. Let S denote the set consisting of these two readings. We explain how to compute the 0-resilient linear envelope of S at time $t_0 = 5$.

A line is in the 0-resilient linear support of S iff it passes through both readings. One such line is shown

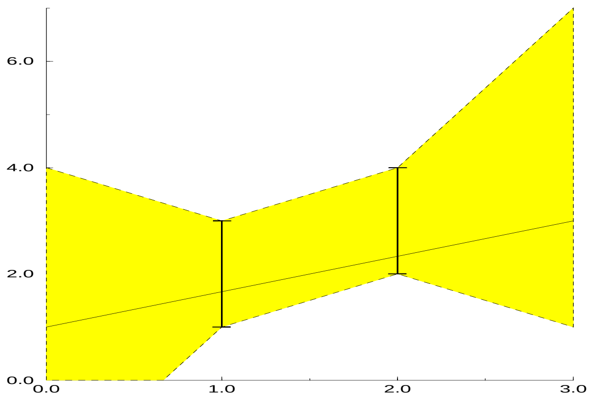

Figure 1: Two readings and a line supported by them

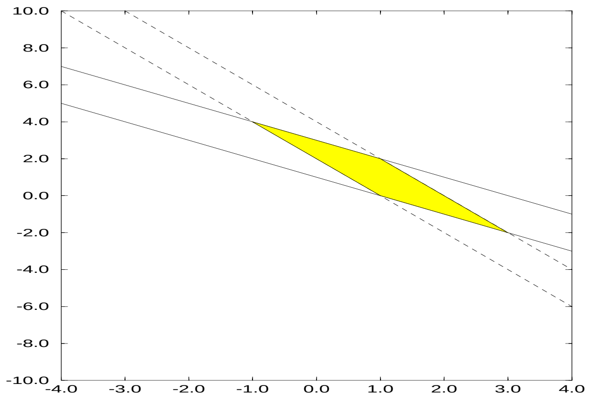

Figure 2: Dual space representation of two readings

in the diagram. A point is in the 0-resilient envelope of S iff it belongs to a line that is in the 0-resilient support of S. Thus, the 0-resilient envelope of S is the union of all lines belonging to the 0-resilient support of S. The 0-resilient envelope is depicted by the shaded region of the diagram.

As can be seen from Figure 1, the 0-resilient envelope. is somewhat complex. A simpler represen tation can be obtained by transforming it to a dual space, where a point (a, b) in the dual space defines a line $y = at + b$ in our original parameter space. Given a reading $(t, [u, v])$ and a point (a, b) in the dual space, the line defined by (a, b) is in the support of $(t, [u, v])$ iff $at + b \in [u, v]$. In other words, (a, b) is in the support of $(t, [u, v])$ iff the following two inequalities are satisfied:

$$\begin{aligned} b &\leq v - at \\ b &\geq u - at \end{aligned}$$

Thus, each reading gives rise to a banded area in the dual space that is the intersection of two half-planes. The 0-resilient support of a collection of readings is the intersection of the respective bands, which is a bounded convex polygon.

Figure 2 shows the support in the dual space that arises from the readings depicted in Figure 1. The region between the solid lines defines the support of r_1, while the region between the dashed lines defines the support of r_2. The shaded region defines the set of points that are supported by both r_1 and r_2.

Each point (a, b) within the shaded region defines a line $y = at + b$ that is supported by both readings. To determine the minim um value that is in the 0-resilient envelope of these two readings at time $t_0 = 5$, it is sufficient to evaluate the function $y = at + b$ at $t_0 = 5$ for each point (a, b) in the shaded region. Evaluation at $t = 5$ defines a linear function on this region, and therefore the minimum value of this function will be attained at one of the vertices of the shaded region. By similar reasoning, the maxim um value will also be attained at one of the vertices.

The coordinates of the vertices and the corresponding values at $t = 5$ are given by the following table:

Vertex	**Value at $t = 5$**
(-1,4)	-1
(1,0)	5
(1,2)	7
(3,-2)	13

From this table, we can see that the smallest interval containing the 0-resilient envelope at time $t = 5$ is $[-1, 13]$.

The situation is more complex when some of the readings may be faulty. In general, a collection of n readings will give rise to a geometric configuration in the dual space defined by $2n$ inequalities. This geometric configuration will consist of a number of vertices, edges and faces, as determined by the intersections of the inequalities.

Consider a face F within the configuration and an inequality $b \leq v_i - at_i$. Either all of the points within F will satisfy the inequality or none of them will. Similarly, either all of the points within F will satisfy the inequality $b \geq u_i - at_i$ or none of them will. If all of the points within F satisfy both inequalities, then F is supported by the reading r_i. We may therefore associate to each face F the number $s(F)$ of readings that support F.

To determine the f-resilient envelope at a time t_0, we begin by computing the corresponding planar configuration. For each face of the configuration that is supported by at least $n - f$ readings, the f-resilient envelope of that face can be determined by scanning each of the vertices of the face. Then, the f-resilient envelope of the entire set of readings can be determined by selecting the minim um and maxim um values that are found in the f-resilient envelopes of each of the faces.

3.2 Algorithm

The algorithm for computing the f-resilient envelope in the linear case is given in Figure 3. We now prove that this algorithm is correct and analyze its running time.

Theorem 1 *Let* $(t_1, [u_1, v_1]), \ldots, (t_n, [u_n, v_n])$ *be a*

Input: A collection $(t_1, [u_1, v_1]), \ldots, (t_n, [u_n, v_n])$ of readings, a distinguished time t_0, and the maximum number f of faulty readings.

Output: The smallest interval $[u, v]$ that contains the f-resilient envelope of the readings at time t_0.

Algorithm:

1. Convert each reading $(t_i, [u_i, v_i])$ to the pair of lines, $b = v - at_i$ and $b = u - at_i$.
2. Compute the planar configuration defined by these $2n$ lines.
3. Label each face F in the configuration by the number $s(F)$ of readings that support it. Let R by the set of faces satisfying $s(F) \geq n - f$.
4. For each face $F \in R$, define $V(F)$ to be the set of vertices of this face. Let $u_F = \min_{(a,b)\in V(F)}\{at_0 + b\}$ and let $v_F = \max_{(a,b)\in V(F)}\{at_0 + b\}$.
5. Output the interval $[u, v]$, where $u = \min_{F\in R}\{u_F\}$ and $v = \max_{F\in R}\{v_F\}$.

Figure 3: Algorithm for fault-tolerant sensor prediction

collection of readings, f be a maximum number of faulty readings, and t_0 be a distinguished time. If $[u, v]$ is the interval computed by the algorithm in Figure 3 on this input, and if (t_0, w_0) is in the f-resilient linear envelope of these readings, then $u \leq w_0 \leq v$.

Proof: Since (t_0, w_0) is in the f-resilient linear envelope of the readings, it follows that there is a line $y = a_0t + b_0$ that is in the f-resilient support of these readings and such that $y(t_0) = w_0$. Let F be the face containing (a_0, b_0) in the planar configuration computed by the algorithm. Since $y = a_0t + b_0$ is in the f-resilient support, there are indices $i_1, i_2, \ldots, i_{n-f}$ such that $y(t_{i_k}) \in [u_{i_k}, v_{i_k}]$ for $k = 1, \ldots, n - f$. Therefore, F is supported by each of the readings $i_1, i_2, \ldots, i_{n-f}$, so we have $s(F) \geq n - f$. Hence, F is one of the faces included in the set R which is computed in Step 3 of the algorithm.

Let $[u_F, v_F]$ be the interval computed in Step 4 of the algorithm. Note that the function $at + b$ is a linear function of (a, b) and that the face F is a convex polygon. Therefore, this function takes its minimum and maximum values at the vertices of F. Consequently, it follows that $u_F \leq a_0t_0 + b_0 = w_0$ and that $v_F \geq a_0t_0 + b_0$.

Note that the values of u and v computed in Step 5 of the algorithm satisfy $u \leq u_F$ and $v_F \leq v$. Therefore, $u \leq u_F \leq w_0 \leq v_F \leq v$. □

Theorem 2 *Let $S = \{(t_1, [u_1, v_1]), \ldots, (t_n, [u_n, v_n])\}$ be a collection of readings, f be a maximum number of faulty readings, and t_0 be a distinguished time. If $[u, v]$ is the interval computed by the algorithm in Figure 3, then the points (t_0, u) and (t_0, v) are in the f-resilient linear envelope of these readings.*

Proof: Let F be a face which satisfies $u_F = u$. Then, F contains a vertex (a, b) such that $at_0 + b = u$. By assumption, $s(F) \geq n - f$, so the line defined by (a, b) is in the f-resilient support of S. Since the point (t_0, u) belongs to this line, it follows that (t_0, u) is in the f-resilient linear envelope of S.

The argument that shows that v is in the f-resilient linear envelope of S is identical. □

Putting these two facts together, we see that the f-resilient envelope computed by our algorithm is contained in the output interval $[u, v]$ and the endpoints of this interval belong to the f-resilient envelope. Therefore, $[u, v]$ must be the smallest interval which contains the f-resilient envelope of S at time t_0. Consequently, we have the following:

Theorem 3 *Let $S = \{(t_1, [u_1, v_1]), \ldots, (t_n, [u_n, v_n])\}$ be a collection of readings, f be a maximum number of faulty readings, and t_0 be a distinguished time. The interval $[u, v]$ computed by the algorithm in Figure 3 is the smallest interval containing the f-resilient envelope of S at time t_0.*

If f is too large, then the interval that is computed by our algorithm may be unbounded. We show that if f is below a certain value, then the interval is bounded.

Theorem 4 *The width of the interval computed by our algorithm is bounded if $n \geq 2(f + 1)$.*

Proof: Assume that $n \geq 2(f + 1)$. Consider any point in the f-resilient envelope. By definition, this point lies on a line that is supported by at least $n - f$ readings. No more than f of these readings can be incorrect, and so the point lies on a line that is supported by at least $n - 2f$ correct readings. Hence, it lies on a line that is supported by at least two correct readings. Thus, the f-resilient envelope is contained within the union of the 0-resilient envelopes of all pairs of correct readings. The 0-resilient envelope of any two readings at a fixed time t_0 is bounded. Since there is a finite number of pairs of correct readings and each corresponding envelope is bounded, the f-resilient envelope at time t_0 is bounded. □

Theorem 5 *The total size of the collection of vertices, edges and faces in any planar configuration of n lines is no more than $O(n^2)$.*

Proof: We show this by enumeration. The number of vertices is at most $\binom{n}{2}$ because each vertex is defined by the intersection of two lines. Each line can be subdivided into at most n edges (since each line is intersected by no more than $n - 1$ lines), and so the number of edges is no more than n^2. Finally, each such edge is shared by two faces, and so the number of faces is no more than twice the number of edges. □

In order to bound the running time of our algorithm, we make use of the following result which was discovered independently by Chazelle et. al. and by Edelsbrunner et. al. [CGL, EOS]. This result appears as Theorem 1 in [CGL] and as Theorem 3.3 in [EOS].

Theorem 6 ([CGL, EOS]) *It is possible to compute an arrangement of n lines in $O(n^2)$ time.*

We are now prepared to bound the running time of the linear prediction algorithm.

Theorem 7 *The algorithm in Figure 3 can be implemented with running time $O(n^2)$.*

Proof:
Step 1 of the above algorithm can clearly be implemented in $O(n)$ time. As we have noted, Step 2 can be implemented in $O(n^2)$ time. We may assume that the data structure produced in Step 2 can be represented as follows:

- Associated to each face F is a list of the edges that bound that face.
- Associated to each edge E is the pair of faces that are bounded by that edge.
- Associated to each edge E is the reading from which that edge was derived.
- Associated to each edge E is the pair of endpoints that define that edge.

Let F and F' be two faces sharing a common edge E. Let ℓ be one of the lines computed in Step 1 of the algorithm. Observe that if E does not belong to ℓ, then both faces are on the same side of ℓ, and if E belongs to ℓ, then the faces are on opposite sides of ℓ.

Let $(t_i, [u_i, v_i])$ be the reading that gave rise to ℓ. If F is supported by $(t_i, [u_i, v_i])$, then F' is not supported by it. On the other hand, if F is not supported by $(t_i, [u_i, v_i])$, then F' is supported by it. Thus, we see that either $s(F') = s(F) - 1$ or $s(F') = s(F) + 1$. Moreover, given the value of $s(F)$, we can determine the value of $s(F')$ simply by testing any vertex of F' not belonging to ℓ for whether or not it is supported by $(t_i, [u_i, v_i])$.

In order to compute the labels $s(F)$ in Step 3, we begin by choosing an arbitrary face F_0 and an arbitrary point that is in the interior of F_0 (being a convex polygon, such a point can be found by taking the average of three vertices of F_0). We then test this point against each of the n input readings. Thus, $s(F_0)$ is determined in time $O(n)$.

Having determined $s(F_0)$, the remarks of the preceding paragraphs show that we can determine $s(F)$ for any neighbor F of F_0 in constant time. Since a depth first traversal of an arbitrary graph can be performed in time proportional to the size of the graph, we can determine $s(F)$ for each face of the configuration in total time $O(n^2)$. Thus, Step 3 can be computed in time $O(n^2)$.

Step 4 can be computed as follows. For each face F that satisfies $s(F) \geq n-f$, we scan the edges bounding F. For each edge E bounding F, we evaluate at_0+b for each endpoint (a, b) of E. We record this value as u_F if it is smaller than any previously encountered value on face F, and we record it as v_F if it is larger than any previously encountered value on face F. Since no edge belongs to more than two faces, no edge is examined more than twice, and since each edge results in the evaluation of two vertices, the total number of times vertices are evaluated is no more than four times the number of edges. Since the number of edges is $O(n^2)$, Step 4 can be completed in time $O(n^2)$.

Finally, Step 5 can be computed by linearly scanning each of the faces satisfying $s(F) \geq n - f$, recording the minimum value of u_F and the maximum value of v_F. Thus, Step 5 can be completed in time $O(n^2)$.

Since each of the steps of this algorithm can be completed in time at most $O(n^2)$, it follows that the algorithm terminates in time $O(n^2)$. □

4 Extensions

In this section, we discuss two ways in which the results of the previous section can be extended. In the direction of obtaining faster algorithms, we consider whether or not we can obtain an efficiency increase by posing the simpler question, "Is a designated point (t_0, u_0) in the f-resilient support of a set S of readings?" We call this question the *f-resilient membership problem.* In general, this question appears to be no easier than determining the f-resilient envelope at time t_0. However, if we view the set S as constant, we can build an efficient algorithm for deciding the membership problem for various query points (t_0, u_0):

Theorem 8 *Let S be a collection of readings, and f be a maximum number of faulty readings. If $O(n^2)$ pre-processing time is allowed, then the membership problem can be solved in time $O(n)$.*

Proof: Given a point (t_0, u_0), we are asked to determine if there is an f-resilient function $y = at + b$ such that $u_0 = at_0 + b$. Viewing t_0 and u_0 as fixed, this equation defines a line ℓ in the dual space. We see that (t_0, u_0) is in the f-resilient envelope iff ℓ impinges upon a face F in the dual space satisfying $s(F) \geq n-f$.

In the pre-processing step of the algorithm, we compute the geometric configuration defined by the readings and label each face F of the configuration with $s(F)$, as described in the previous section. This step can be performed in time $O(n^2)$.

The method given by [CGL] is incremental, in the sense that the geometric configuration is computed by successively inserting lines into the configuration. Each insertion operation takes time $O(k)$, where k is the number of lines which have been previously inserted into the data structure.

To determine the set of faces intersected by ℓ, we need only simulate an insertion operation in such a way that the set of faces impinged by ℓ can be recorded and so that the underlying data structure is not modified. This can be done in time $O(n)$ and so our claim is verified. □

Another direction in which these results can be extended is through considering supporting functions that are degree-d polynomials. We say that a function $y(t)$ is in the *f-resilient degree-d support* of S if $y(t)$ is in the f-resilient support of S and $y(t)$ is a polynomial of degree at most d. Similarly, we define the *f-resilient degree-d envelope* of S to be the set of points which satisfy functions in the f-resilient degree-d support of S.

In order to analyze this problem, we make use of the following result which appears as Theorem 3.3 from [EOS].

Theorem 9 *Let H be a set of n hyperplanes in E^d, for $d \geq 2$. Then there is an algorithm for computing the corresponding geometric arrangement in time $O(n^d)$.*

Theorem 10
Let $S = \{(t_1, [u_1, v_1]), \ldots, (t_n, [u_n, v_n])\}$ be a collection of readings, f be a maximum number of faulty readings, and t_0 be a distinguished time. The smallest interval containing the f-resilient degree-d support of S at time t_0 can be computed in time $O(n^{d+1})$.

Proof: Consider a polynomial $y(t) = a_d t^d + a_{d-1}t^{d-1} + \cdots + a_0$. This polynomial is supported by a reading $(t_i, [u_i, v_i])$ iff $u_i \leq y(t_i) \leq v_i$. For each reading, we may let

$$\begin{aligned} U_i &= \{(a_0, a_1, \ldots, a_d) \mid a_d t_i^d + a_{d-1} t_i^{d-1} + \cdots a_0 \geq u_i\} \\ V_i &= \{(a_0, a_1, \ldots, a_d) \mid a_d t_i^d + a_{d-1} t_i^{d-1} + \cdots a_0 \leq v_i\} \end{aligned}$$

From this representation, we can see that U_i and V_i are halfspaces in $(d+1)$-dimensional Euclidean space. By the results of [EOS], it is possible to compute the $(d+1)$-dimensional geometric configuration defined by these sets in time $O(n^{d+1})$. The values of $s(F)$, u_F and v_F for each of the $(d+1)$-dimensional faces F in this configuration can be computed using the same method as in the two dimensional case. $\square$

5 Conclusions

In this paper, we have presented a method for predicting future values of a physical variable, given imprecise and possibly faulty measurements of its past values. We make no statistical assumptions about the distribution of errors, except that out of n such measurements no more than f can be incorrect.

Our work was motivated by the observation that clocks in a distributed system tend to drift apart at a constant rate. We were led to ask whether or not this observation could be used to achieve tighter synchronization than that possible with earlier protocols. Other protocols for clock synchronization make the assumption that clocks drift at a bounded rate. While this assumption allows synchronization to be achieved, old clock values quickly become useless because the imprecision of a clock reading is magnified over time.

The approach in this paper shows how the bounded drift rate assumption can be traded for an alternative hypothesis, namely that the drift between clocks is approximately linear for sufficient periods of time. Under this assumption, old clock values retain their usefulness, as they are implicitly used to determine the true drift rate between clocks.

We have recently discovered an $O(n^2 \log n)$ algorithm for the linear prediction problem which computes the solution directly from the input readings, without need for computing the planar configuration. Consequently, even though this algorithhm has worse asymptotic complexity, it is much faster for reasonable values of n. We are using this algorithm in the clock synchronization component of a fault-tolerant real-time toolkit which we are building.

We are hopeful that the techniques outlined in this paper will find application in other control systems where the underlying physical process can be described by a linear function or low degree polynomial.

References

[CM] Clegg, M. and K. Marzullo. Clock Synchronization in Hard Real-Time Distributed Systems. University of California, San Diego Department of Computer Science and Engineering Technical Report CS96-478, February 1996.

[DLR] Dempster, A. P., N. M. Laird and D. B. Rubin. Maximum liklihood from incomplete data via the EM algorithm. *J. Royal Stat. Soc. Series B* 39 (1977) 1-22.

[CGL] Chazelle, B.M., L.J. Guibas, and D.T. Lee. The power of geometric duality. *BIT* 25 (1985) 76-90.

[EOS] Edelsbrunner, H., J. O'Rourke and R. Seidel. Constructing arrangements of lines and hyper planes with applications. *SIAM J Computing* 15, 1986, 341-63.

[M] Marzullo, K. Tolerating failures of continuous-valued sensors. *ACM Transactions on Computer Systems* Nov. 1990, vol. 8, (no. 4): 284-304.