Journal of Intelligent Information Systems 13, 47–71 (1999)

Deriving Trends in Historical and Real-Time Continuously Sampled Medical Data

APKAR SALATIAN as@scms.rgu.ac.uk
School of Computer and Mathematical Sciences, The Robert Gordon University, St. Andrew Street, Aberdeen AB25 1HG, UK

JIM HUNTER jhunter@csd.abdn.ac.uk
Department of Computing Science, King's College, University of Aberdeen, Aberdeen AB24 3UE, UK

Abstract. Monitors in Intensive Care Units generate large volumes of continuous data which can overwhelm a database and result in information overload for the medical staff. Instead of reasoning with individual data samples of one or more variables, it is better to work with the trend of the data i.e., whether the data is *increasing*, *decreasing* or *steady*. We have developed a system which abstracts continuous data into trends; it consists of three consecutive processes: *filtering* which smooths the data; *temporal interpolation* which creates simple intervals between consecutive data points; and *temporal inference* which iteratively merges intervals which share similar characteristics into larger intervals. Storing trends can result in a reduction in database volume. Our system has been applied both to historical and real-time data.

Keywords: intensive care, interval identification, temporal interpolation, temporal inference

1. Introduction

In many domains we are confronted with large sets of continuously sampled data. Reasoning about the relationships between the consecutive individual measurements of one variable is computationally expensive—and gets worse when several variables are interpreted together. We agree with (Kohane and Haimowitz, 1993) when they say that: 'The abstraction of primary data into intervals over which a specified predicate holds is a central task in process monitoring. It relieves the monitoring program of the complexity of having to repeatedly reason about the relationships between each datum in potentially vast data sets.' In some domains a number of variables are measured and the interval between data samples can be as little as a few seconds. Such large amounts of data can overwhelm a database so by abstracting intervals over a group of data points we can reduce the volume of stored data.

In this report we propose an algorithm for deriving temporal intervals over which the attribute with possible values *steady* or, with different rates of change (slow, moderate or rapid), *increasing* or *decreasing* holds; such an attribute will be called the *trend* of the data over the interval. In figure 1 for example, we might want to say that between t_1 and t_2 a signal was steady, and between t_2 and t_3 it was rapidly *increasing*, and so on. It is important to note that the end points of the intervals are not given in advance; they have to be discovered by the analysis of the raw data.

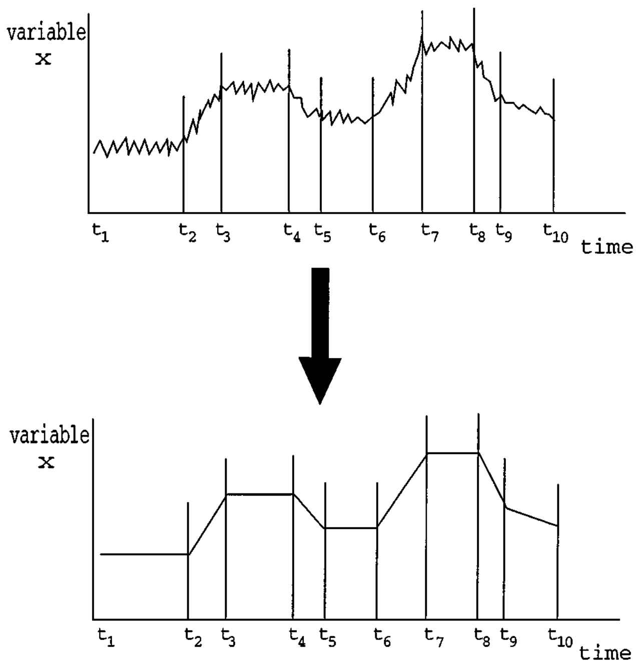


Figure 1. Generating temporal intervals from data points.

In considering time series data, two perspectives are possible:

- *Historical:* By this we mean that in analysing the data at a time t, we have access to data both in the past and in the future relative to t.
- *Real time:* In this case, data is being added to the series as time passes, and our interest is in developing a description of what is happening at that ‘leading edge’. We are not concerned here with the absolute speed of the algorithms—i.e., we do not worry whether they could keep up with the incoming data. We have only the constraint that when processing the most recent data point, we have access to data which is relatively in the past.

There is much in common in the two situations; the differences will be made clear. Temporal data abstraction is only part of the task of intelligent signal analysis. Abstractions can be used as the input to pattern matchers leading to higher levels of interpretation, can be displayed to the user as graphical summaries or can be summarised textually. The abstraction techniques reported here have been used in conjunction with a form of trend template (Haimowitz and Kohane, 1996) on ICU data for data validation (removal of artefacts, e.g., transcutaneous probe change) and the identification of abnormal clinical conditions (e.g., pneumothorax) (Salatian, 1997).

Our approach relies on an initial pre-filtering to smooth the data which is then passed to two further processes which generate the intervals. The structure of this paper is as follows.

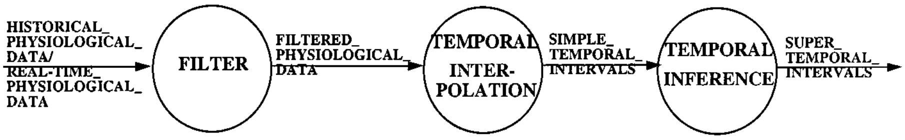


Figure 2. The processes which are used to identify intervals.

Section 2 describes our algorithm for deriving intervals from historical data; Section 3 contains the modifications necessary to handle the data in real time. Section 4 presents the results of applying these processes to various data sets. Section 5 describes related work in trend detection. Finally, Section 6 summarises and concludes this report.

2. Interval identification in historical data

This process can be divided into the sub-processes *pre-filtering*, *temporal interpolation* and *temporal inference*—see figure 2. In what follows we will be constructing temporal intervals; the ith temporal interval, I_i, is described as:

$$I_i(x, t_{\text{begin}}, t_{\text{end}}, trend_i, \alpha_i, \beta_i, \lambda_i, \mu_i)$$

where

- x is the variable under investigation (e.g., heart rate)
- t_{begin} is the start time of the interval
- t_{end} is the end time of the interval
- $trend_i$ is the interval trend i.e., *increasing*, *decreasing* or *steady*
- α_i is the minimum value of x over the interval
- β_i is the maximum value of x over the interval
- λ_i is the absolute value of the gradient of x over the interval
- μ_i is the mean of x over the interval

All of the numerical properties of the interval are calculated simply from the values of the data points which it encompasses. The issues in question are (i) how to determine the extent of the individual intervals and (ii) how to determine the trend.

2.1. Pre-filtering

We investigated the following filters: *low-pass*, *high-pass*, *band-pass*, *median* and *average*. All of these techniques involve a moving window. For historical data, the window can be centered on the point x_n i.e., if the window is of size $2k + 1$ the window contains the points x_{n-k} to x_{n+k}. For real-time data the approaches are similar, but the current data point is at the leading edge of the window which contains points x_{n-2k} to x_n. In what follows we will only consider centered windows.

After various investigations, a median filter was chosen. Median filtering is a non-linear signal processing method which replaces the input sequence with the running median over a window of some specified length (Haddad and Parsons, 1991). Given a window length of $2k + 1$, let the function median$(x_{n-k}, \ldots, x_n, \ldots, x_{n+k})$ represent the median of the input values from x_{n-k} to x_{n+k}. Then the output, y_n, for an input point x_n is:

$$y_n = \text{median}(x_{n-k}, \ldots, x_n, \ldots, x_{n+k})$$

A brute-force implementation of a median filter simply copies the values from x_{n-k} to x_{n+k}, sorts the copied values, and outputs the central point in the sorted sequence. Median filtering can remove both noise and transients from the signal without distortion of the base line. It will remove transient features lasting shorter than half the width of the window; features lasting more than half the width of the window will remain. Figure 3 shows the effect of applying a median filter with $k = 10$ (window size 21)—this length of window is suitable because we are interested in features in the data lasting more than 10 min. It can be seen that a median filter has a very good performance at sudden, step like changes of a signal—they are exactly reproduced.

For real time data, exactly the same effect is obtained when the current point is at the leading edge of the window, except that the entire curve is shifted back by an amount k.

For a full description of the properties of median filters, the reader is referred to (Gallagher and Wise, 1981).

2.2. *Temporal interpolation*

This is the process of generating an interval between two adjacent data points. By ‘adjacent’ we mean that there are no missing data points between them. Note that in all our raw data, 0 (zero) may stand for a true zero data point or may stand for missing data; there is no way of knowing which. We will refer to the value of x at t_{begin} as x_{begin} and to the value of x at t_{end} as x_{end}.

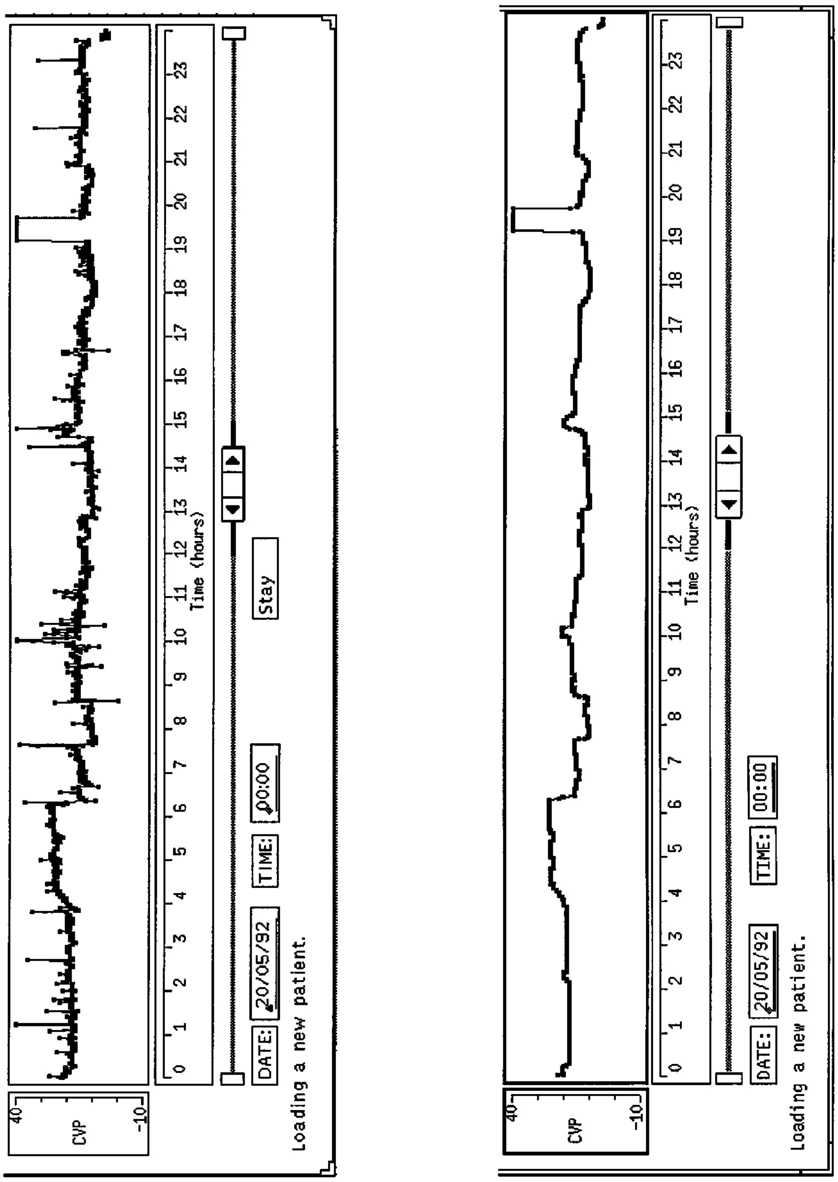
The trend *steady* ($trend_i = steady$) is derived if $x_{\text{end}} = x_{\text{begin}} \pm \delta x$. The value of δx will depend on the resolution of the instrument supplying the data. In our case the level of quantisation is such that we can take it to be zero (i.e., we apply strict equality). If the interval is not classified as *steady*, then it will be *increasing* if $x_{\text{end}} > x_{\text{begin}}$ and *decreasing* if $x_{\text{end}} < x_{\text{begin}}$.

The values of α_i, β_i and μ_i are calculated as follows:

- $\alpha_i = \min(x_{\text{end}}, x_{\text{begin}})$
- $\beta_i = \max(x_{\text{end}}, x_{\text{begin}})$
- $\mu_i = \frac{(x_{\text{end}} + x_{\text{begin}})}{2}$

For *steady* intervals λ_i, is set to zero. For increasing or decreasing intervals it is defined as $\frac{|x_{\text{end}} - x_{\text{begin}}|}{t_{\text{end}} - t_{\text{begin}}}$.

Temporal interpolation involves a single pass over the data set. Given n data points, it generates exactly $n - 1$ *simple temporal intervals*.



(a) (b)

Figure 3. Application of a median filter: (a) original signal; (b) signal after applying a median filter.

2.3. *Temporal inferencing—general considerations*

Interviews were performed with clinicians to determine how trends in the data are derived. Clinicians are interested in whether trends are steady, increasing or decreasing, and on the data's rate of change.

An interval is steady if the difference between any two points in the interval is below a threshold. This threshold will determine maximum and minimum values for the interval. The threshold is dependent on the specific variable and is defined by clinicians. The value of a threshold depends on the variability of the variable. The greater its variability, the larger the threshold. Likewise if there is little variability in the data values then the variability may be considered small. For example, we may consider a heart rate to be steady if there is no deviation of more than 5 beats per min i.e., the difference between the maximum and minimum values for the steady interval is less than or equal to 5 beats per min.

A deviation greater than the allowable range for a *steady* may be considered to be either an increasing or decreasing trend. Clinicians are interested in different rates of change, namely whether an increasing or decreasing trend is slow, moderate or rapid—these trends are defined by different gradient ranges and will depend on the particular variable and the context in which the abstraction takes place. Such information allows clinicians to identify specific clinical conditions. For example anaphylactic shock is determined by a rapidly increasing heart rate, rapidly decreasing blood pressure and rapidly decreasing central venous pressure.

Temporal Inferencing is the process of attempting to apply rules to merge two or three meeting intervals into *super-intervals*, so that a common trend can be derived. The rules for merging must take account of the fact that a super-interval which is described, for example, as increasing, may actually be made up of some smaller sub-intervals in which the variable is described as steady or even decreasing.

Such rules are based on *duration* and *rate of change*. For example, given three intervals which are increasing, steady and increasing respectively one can infer a possible increasing super-interval if the duration of the steady interval is shorter than the duration of the increasing intervals. Likewise a *steady* interval may be made up of many increasing, decreasing and steady intervals. Here one needs upper and lower thresholds and an appropriate metric is provided by the minimum and maximum values of the steady interval.

To derive super-intervals, we define the following four parameters for each variable:

- dur = a duration
- $diff$ = a small range; used in the definition of *steady*
- g_1 = a gradient (see below)
- g_2 = a gradient, greater than g_1 (see below)

If a gradient is less than or equal to g_1 ($\lambda_i \leq g_1$) this is taken as representing a *slow* rate of change. If it is between g_1 and g_2 ($g_1 < \lambda_i \leq g_2$) this is taken as representing a *moderate* rate of change. If it is greater than g_2 ($g_2 < \lambda_i$) this is taken as representing a *rapid* rate of change. If the application requires, a different number of qualitative rates of change can be defined.

As intervals are merged into super-intervals, a new data structure is created representing the new interval, derived from the representations of its sub-intervals.

Temporal inferencing is done in two ways:

- over *two* meeting intervals
- over *three* meeting intervals.

Based on similar characteristics we merge the simple intervals generated from the temporal interpolation process into larger intervals then repeatedly merge these larger intervals into even larger intervals until no more similarities can be found. This merging algorithm is achieved using the temporal inference rules. Firstly we apply rules to merge two meeting intervals to derive only increasing and decreasing trends—this will provide the basis for finding potentially larger increasing and decreasing trends. The term ‘apply’ means we try to combine the first two intervals; if this succeeds then try to combine this new interval with the next and so on. If two meeting intervals cannot be merged then we take the second of the intervals which were to be combined, and use it as a starting point. We then apply rules to merge two meeting intervals which are either increasing and decreasing, decreasing and increasing or both steady to derive only steady intervals. We then repeatedly apply the inferences to merge three meeting intervals followed by the inferences to merge two meeting intervals which derive only steady intervals until no more intervals can be merged.

2.4. Temporal inferencing over two meeting intervals

For temporally inferencing over two meeting intervals we want to represent a *super-interval* as an assertion in the following form:

$$\Delta_{H2}(I_i(x, t_1, t_2, \mathit{trend}_i, \alpha_i, \beta_i, \lambda_i, \mu_i), I_j(x, t_2, t_3, \mathit{trend}_j, \alpha_j, \beta_j, \lambda_j, \mu_j))$$
$$\Rightarrow I_{ij}(x, t_1, t_3, \mathit{trend}_{ij}, \alpha_{ij}, \beta_{ij}, \lambda_{ij}, \mu_{ij})$$

Thus given intervals I_i (from t_1 to t_2) and I_j (from t_2 to t_3) which meet at t_2, we can derive a *super-interval* I_{ij} (from t_1 to t_3) by merging the intervals I_i and I_j based on the temporal inferencing function Δ_{H2}.

Table 1 shows the possible combinations of trends for two meeting intervals and the possible super-interval which can be derived, provided that further criteria are satisfied.

By symmetry, *decreasing* followed by *decreasing* is similar to *increasing* followed by *increasing*. The criterion for combining any two intervals into a steady interval is independent of the trends of the two contributing intervals. We have therefore only two distinct cases to consider.

Whenever a super-interval is generated, the values of α_{ij}, β_{ij} and μ_{ij} are calculated as follows:

- $\alpha_{ij} = \min(\alpha_i, \alpha_j)$
- $\beta_{ij} = \max(\beta_i, \beta_j)$

Table 1. Inference table for two meeting intervals.

Δ_{H2}	$I_j(x, t_2, t_3, \mathit{incr}, \ldots)$	$I_j(x, t_2, t_3, \mathit{std}, \ldots)$	$I_j(x, t_2, t_3, \mathit{decr}, \ldots)$
$I_i(x, t_1, t_2, \mathit{incr}, \ldots)$	$I_{ij}(x, t_1, t_3, \mathit{incr}, \ldots)$	$I_{ij}(x, t_1, t_3, \mathit{std}, \ldots)$	$I_{ij}(x, t_1, t_3, \mathit{std}, \ldots)$
$I_i(x, t_1, t_2, \mathit{std}, \ldots)$	$I_{ij}(x, t_1, t_3, \mathit{std}, \ldots)$	$I_{ij}(x, t_1, t_3, \mathit{std}, \ldots)$	$I_{ij}(x, t_1, t_3, \mathit{std}, \ldots)$
$I_i(x, t_1, t_2, \mathit{decr}, \ldots)$	$I_{ij}(x, t_1, t_3, \mathit{std}, \ldots)$	$I_{ij}(x, t_1, t_3, \mathit{std}, \ldots)$	$I_{ij}(x, t_1, t_3, \mathit{decr}, \ldots)$

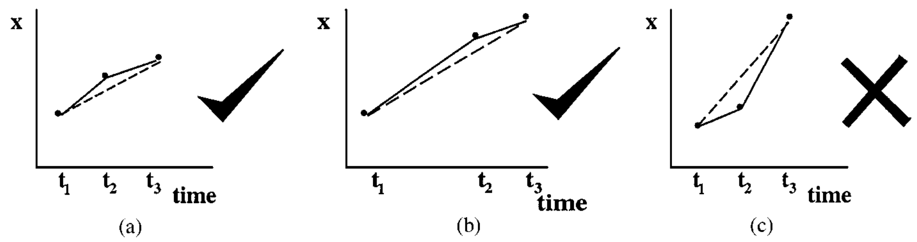


Figure 4. Inferring over two meeting *increasing* intervals.

- $\mu_{ij} = \frac{\mu_i * (t_2 - t_1) + \mu_j * (t_3 - t_2)}{t_3 - t_1}$
- λ_{ij} depends on $trend_{ij}$; if $trend_{ij}$ is *steady* then $\lambda_{ij} = 0$; if $trend_{ij}$ is *increasing* or *decreasing* then $\lambda_{ij} = \frac{|\beta_{ij} - \alpha_{ij}|}{t_3 - t_1}$

2.4.1. Increasing/increasing ⇒ increasing. For successive increasing intervals, a super-interval *increasing* is inferred if the gradients of *both* intervals are within the *same* limit range i.e., only intervals with the same rate of change (slow, moderate or rapid) are combined. Formally we can write this as:

$\lambda_i \leq g_1$ **AND** $\lambda_j \leq g_1$
OR
$g_1 < \lambda_i \leq g_2$ **AND** $g_1 < \lambda_j \leq g_2$
OR
$g_2 < \lambda_i$ **AND** $g_2 < \lambda_j$

If this condition is not satisfied, no inference can be performed.

A typical situation is illustrated in figure 4(a)—the dashed line represents the new gradient for the super-interval I_{ij}.

From figure 4(b) it can be seen that we can combine intervals of different durations provided that their gradients are in the same range.

Figure 4(c) shows that inferring over two increasing intervals whose gradients are in different gradient limit ranges would result in a super-interval which had a gradient that did not capture the different rate of change in the data; such a combination is therefore prohibited.

Note that we do not merge successive increasing intervals into a super-interval *increasing* regardless of its rate—though the overall data is increasing we need to capture different rates so that higher level processes can interpret them.

2.4.2. Any/any ⇒ steady. A super-interval is classified as *steady* if the difference between the maximum and minimum values over the interval is less than the pre-defined constant *diff*.

$$\beta_{ij} - \alpha_{ij} <= diff$$

Table 2. Inference table for three meeting intervals which begin with an *increasing* interval.

Δ_{H3}	$I_k(x, t_3, t_4, incr, \ldots)$	$I_k(x, t_3, t_4, std, \ldots)$	$I_k(x, t_3, t_4, decr, \ldots)$
$I_i(x, t_1, t_2, incr, \ldots)$, $I_j(x, t_2, t_3, incr, \ldots)$	$I_{ijk}(x, t_1, t_4, incr, \ldots)$	No merging	No merging
$I_i(x, t_1, t_2, incr, \ldots)$ $I_j(x, t_2, t_3, std, \ldots)$	$I_{ijk}(x, t_1, t_4, incr, \ldots)$	No merging	No merging
$I_i(x, t_1, t_2, incr, \ldots)$ $I_j(x, t_2, t_3, decr, \ldots)$	$I_{ijk}(x, t_1, t_4, incr, \ldots)$	No merging	No merging

2.5. *Temporal inferencing over three meeting intervals*

Inferring over two meeting intervals alone can result in too many intervals. Inferring over three meeting intervals in particular cases allows us to create even larger intervals.

For three meeting intervals we want to represent a super-interval as an assertion in the following form:

$$\begin{aligned}&\Delta_{H3}(I_i(x, t_1, t_2, trend_i, \alpha_i, \beta_i, \lambda_i, \mu_i), I_j(x, t_2, t_3, trend_j, \alpha_j, \beta_j, \lambda_j, \mu_j),\\&\quad I_k(x, t_3, t_4, trend_k, \alpha_k, \beta_k, \lambda_k, \mu_k))\\&\Rightarrow I_{ijk}(x, t_1, t_4, trend_{ijk}, \alpha_{ijk}, \beta_{ijk}, \lambda_{ijk}, \mu_{ijk})\end{aligned}$$

Thus given intervals I_i (from t_1 to t_2), I_j (from t_2 to t_3) and I_k (from t_3 to t_4) a super-interval I_{ijk} beginning at time t_1 and ending at time t_4 can be created by merging I_i, I_j and I_k using the temporal inferencing function Δ_{H3}. Table 2 shows the additional possible combinations of abstractions for three meeting intervals where the first is characterised as *increasing*. The possibilities when the first interval is *decreasing* follow by symmetry. There are no additional possibilities when the first interval is *steady* since it is assumed that they will have been covered in the two interval case.

Though the inferencing is very similar to Table 1, it will be seen that the size of the middle interval is critical in deciding whether to generate a super-interval. Again we want to create super-intervals which truly reflect rates of change and capture trends.

Considerations of symmetry allow us to consider the following three groups of possibilities for trends of three adjacent intervals:

- **increasing/increasing/increasing**
  decreasing/decreasing/decreasing
- **increasing/steady/increasing**
  decreasing/steady/decreasing
- **increasing/decreasing/increasing**
  decreasing/increasing/decreasing

Where there are multiple possibilities, we will consider only the first, the remainder following from symmetry.

Whenever a super-interval is generated, the values of α_{ijk}, β_{ijk} and μ_{ijk} are calculated as follows:

- $\alpha_{ijk} = \min(\alpha_i, \alpha_j, \alpha_k)$
- $\beta_{ijk} = \max(\beta_i, \beta_j, \beta_k)$
- $\mu_{ijk} = \frac{\mu_i*(t_2-t_1)+\mu_j*(t_3-t_2)+\mu_k*(t_4-t_3)}{t_4-t_1}$
- λ_{ijk} depends on $trend_{ijk}$; if $trend_{ijk}$ is *steady* then $\lambda_{ijk} = 0$; if $trend_{ijk}$ is *increasing* or *decreasing* then $\lambda_{ijk} = \frac{|\beta_{ijk}-\alpha_{ijk}|}{t_4-t_1}$

2.5.1. Increasing/increasing/increasing ⇒ increasing. Given three meeting intervals which are all increasing then we will infer an *increasing* super-interval if the duration of the middle interval is less than the duration of the other two intervals by at least a factor of *dur* (a real number, including fractions) and the gradients of the two outside intervals are both within the same range:

$t_3 - t_2 \leq \min((t_2 - t_1), (t_4 - t_3))/dur$
AND
  $\lambda_i \leq g_1$ **AND** $\lambda_k \leq g_1$
  OR
  $g_1 < \lambda_i \leq g_2$ **AND** $g_1 < \lambda_k \leq g_2$
  OR
  $g_2 < \lambda_i$ **AND** $g_2 < \lambda_k$

This is illustrated in figures 5(a) and (b)—note that the dashed line represents the new gradient for the super-interval.

Figure 5(c) shows that if the gradients of the intervals are in different ranges then inferring an *increasing* super-interval would result in an overall gradient which did not capture the different rates of change.

2.5.2. Increasing/steady/increasing ⇒ increasing. Given three meeting intervals which are *increasing*, *steady* and *increasing* respectively, one can infer an *increasing* super-interval if the duration of the steady interval is less than the size of it's neighbouring increasing intervals by at least a factor of *dur* and the gradients of the increasing intervals are both

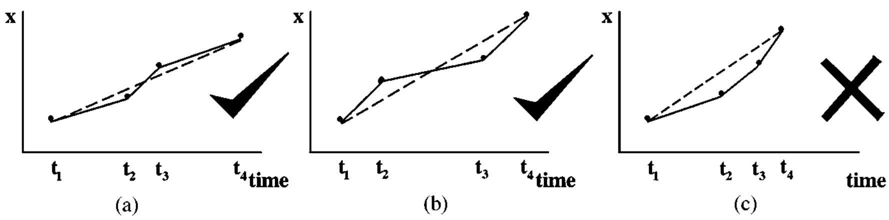


Figure 5. Inferring over three meeting *increasing* intervals.

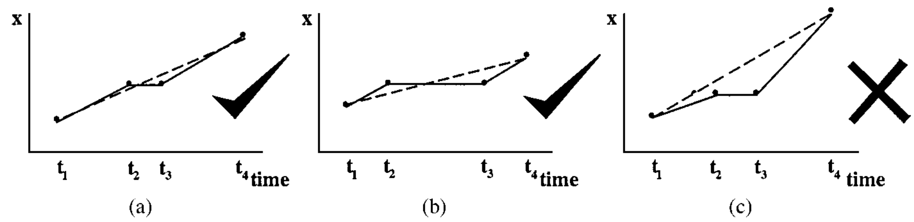


Figure 6. Inferring over meeting *increasing, steady, increasing* intervals.

within the same range. The rule is the same as that given in Section 2.5.1. This is illustrated in figures 6(a) and (b).

If the gradients of the increasing intervals are within different ranges then creating a super-interval which is increasing would result in a gradient which did not reflect the increasing trend. This can be seen in figure 6(c).

2.5.3. Increasing/decreasing/increasing $\Rightarrow$ increasing. Given three meeting intervals which are increasing, decreasing and increasing respectively, one can infer an *increasing* super-interval if the duration of the decreasing interval is less than the duration of it's neighbouring increasing intervals by at least a factor of *dur*, the gradients of the increasing intervals are both within the same range, the minimum value of the decreasing interval (I_j) is greater than the minimum value of the first increasing interval (I_i) and the maximum value of the second increasing interval (I_k) is greater than the maximum value of the first increasing interval. The rule is the same as that give in Section 2.5.1 with the addition of the following conjunctive condition:

$$\alpha_i < \alpha_j \textbf{ AND } \beta_k > \beta_i$$

This is illustrated in figures 7(a) and (b).

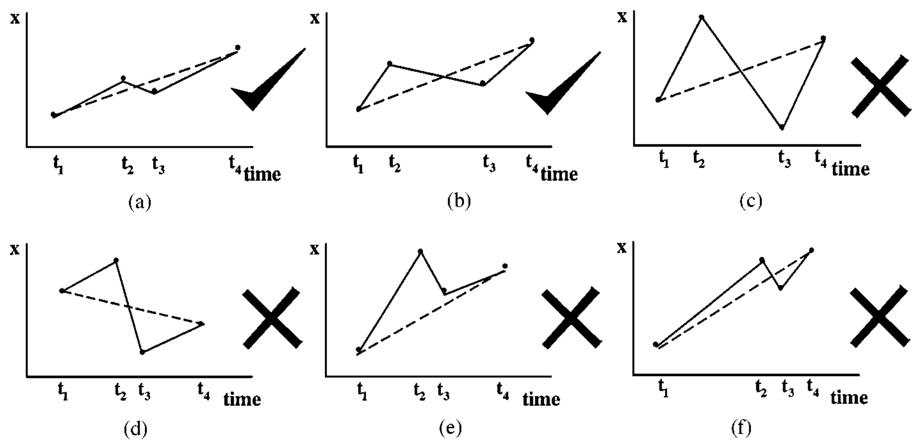


Figure 7. Inferring over meeting *increasing, decreasing, increasing* intervals.

If the minimum value of the decreasing interval (α_j) is less than the minimum value of the first increasing interval (α_i) then inferring a new increasing interval would result in a gradient which is smaller than the gradients of the increasing intervals. This is illustrated in figure 7(c).

If the minimum value of the decreasing interval (α_j) is less than the minimum value of the first increasing interval (α_i) and the maximum value of the second increasing interval (β_j) is less than the minimum value of the first increasing interval (α_i) then inferring a new increasing interval would be equivalent to inferring an interval which is decreasing. This is illustrated in figure 7(d).

If the maximum value of the second increasing interval (β_j) is less than the maximum value of the first increasing interval (β_i) then inferring a new increasing interval would result in a gradient which did not reflect the overall increasing trend. This is illustrated in figure 7(e).

If the duration of the second increasing is not relatively longer than the decreasing interval then inferring an overall increasing interval would result in a smaller gradient which did not reflect the overall increasing trend. This is illustrated in figure 7(f).

2.6. Algorithm

1. Apply the inferences in Δ_{H2} which derive only increasing or decreasing trends. ‘Apply’ means try to combine the first two intervals; if this succeeds then try to combine new interval with the next and so on. If the combination fails, then take the second of the intervals which were to be combined, and use it as a starting point.
2. Apply the inferences in Δ_{H2} which derive steady trends where I_i and I_j are either increasing and decreasing, decreasing and increasing or both steady.
3. Set flag *still-to-do* to **true**.
4. **while** *still-to-do* **do**
5. Set *number-on-last-iteration* to the number of intervals generated so far.
6. Apply the inferences in Δ_{H3}. Like Δ_{H2}, applying function Δ_{H3} is recursive in that the list of intervals is analysed until Δ_{H3} cannot be applied to any three meeting intervals.
7. Apply the inferences in Δ_{H2} which derive only steady trends.
8. Set *still-to-do* to *number-on-last-iteration* $\neq$ current number of intervals.
9. **endwhile**

We can compute the best-case and worst-case time complexity our algorithm takes to create superintervals.

The best-case time complexity of our algorithm is when we capture a single trend over the entire data set by either step 1 or step 2 above. Given *n* simple intervals and assuming step 1 captures the single trend, step 1 would process *n* intervals to create the single interval—this in turn would result in steps 2, 6 and 7 each processing one interval. Thus the best-case time complexity of our algorithm is $O(n + 3)$, or simply $O(n)$.

The worst-case time complexity of our algorithm is when steps 1 and 2 cannot merge any intervals and steps 6 and 7 of the while loop each creates one new super-interval on each iteration resulting in either one or two trends over the entire data set. Given n simple intervals, steps 1 and 2 would each process n intervals and steps 6 and 7 of the while loop would process n and $n-2$ intervals on the first iteration, $n-3$ and $n-5$ on the second etc. Thus the worst-case time complexity of our algorithm is $O(2n + \frac{n}{3}(n+1))$ if n or $(n+1)$ is a multiple of 3 or $O(2n + \frac{n}{3}(n+1) + \frac{1}{3})$ if $(n+2)$ is a multiple of 3, or simply $O(n^2)$.

3. Interval identification in real time data

Pre-filtering of real time data is identical to that of historical data, except that the value generated by the filtering operation is assigned to the time corresponding to the right-hand edge of the window; this introduces an effective delay of k points in a window of length $2k+1$.

In the previous section we presented a technique for merging intervals in historical data. In that case the data set is static; we iterate over the data, merging intervals where possible, until no further merging can be achieved. In real time we have the problem that the data set is continuously being added to; decisions have to be made depending on our current interpretation, in particular on the trend (*increasing*, *steady* or *decreasing*) which we apply to the latest data. This leads to an inevitable tension. If the latest data indicates a change in the trend, then we need to take account of this. However it might just be a momentary aberration, and when more data comes in this will be apparent. The question arises as to how long we can wait before deciding that the new data really does represent a new trend. If this time is too long, we can have more confidence in our interpretation but the time for acting on our new knowledge may have passed. On the other hand, if this time is too short, we may make decisions based on an incorrect interpretation. There is no easy solution to this problem. Our approach is as follows.

Consider as an example figure 8; imagine that the current time is between t_2 and t_3. The data in the past (between t_1 and t_2) belongs to an interval which has an unambiguous trend

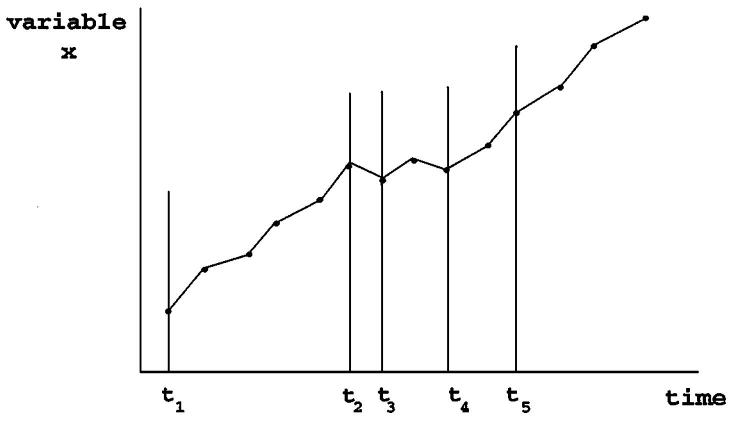


Figure 8. Detecting an overall increasing trend.

(in this case *increasing*; this interval will be referred to as the *previous* interval—I_p. A new data point arrives at t_3. An interval is created by temporal interpolation (see Section 2.2) between the new point and the end of I_p. This new interval will be called the *current* interval—I_c. If I_c is similar enough to I_p (i.e., it satisfies certain criteria which we will define later), it can be merged with I_p to create a *new previous interval*—I_{np}, and we then wait for the next data point. If however, it shows a different trend (as in the example) we don't know whether this means a significant change in the data or whether it is a short term effect which will need to be ignored. As discussed above, time is needed to resolve this so we create a 'delay period' by setting a point in the future by which a decision has to be made (the decision point); in the example this is t_5. We carry on adding in new data points as they arrive. For each simple interval thus generated, we try to merge it with earlier intervals using the approach of Section 2 i.e., trying to merge two or three meeting intervals. The main difference is that the I_p is not 'decomposed' in any way—it can only be extended. As a result of merging these new intervals, it may happen that I_p *is* extended; effectively the interval which caused the deviation was indeed a temporary aberration which can now be accommodated as an extension to I_p. In this case a new decision point is established relative to the end of I_{np}. If, on the other hand, we reach the decision point without any extension, then the last interval to be established is taken to be I_p.

At all times, even when we are in a delay period, the trend is that attached to the previous interval.

In deciding whether to incorporate the current interval into the previous one, we use a set of criteria which are similar to those used to decide whether to merge two intervals with historical data. Again we want to represent a super-interval as an assertion in the following form:

$$\Delta_{R2}(I_p(x, t_1, t_2, trend_p, \alpha_p, \beta_p, \lambda_p, \mu_p), I_c(x, t_2, t_3, trend_c, \alpha_c, \beta_c, \lambda_c, \mu_c))$$
$$\Rightarrow I_{np}(x, t_1, t_3, trend_{np}, \alpha_{np}, \beta_{np}, \lambda_{np}, \mu_{np})$$

Thus given a previous interval I_p (from t_1 to t_2) and the current interval I_c (from t_2 to t_3) which meet at t_2, we can derive a *new previous* super-interval I_{np} (from t_1 to t_3) by merging the intervals I_p and I_c based on the temporal inferencing function Δ_{R2}.

Table 3 shows the possible combinations of trends for the previous and current intervals and the possible super-interval which can be derived, *provided that further criteria are satisfied*. These criteria are set out below.

Considerations of symmetry again lead us to consider the following three groups of possibilities for trends of two adjacent intervals:

Table 3. Inference table for two meeting intervals.

Δ_{R2}	$I_c(x, t_2, t_3, incr, \ldots)$	$I_c(x, t_2, t_3, std \ldots)$	$I_c(x, t_2, t_3, decr \ldots)$
$I_p(x, t_1, t_2, incr, \ldots)$	$I_{np}(x, t_1, t_3, incr, \ldots)$		
$I_p(x, t_1, t_2, std \ldots)$	$I_{np}(x, t_1, t_3, std \ldots)$	$I_{np}(x, t_1, t_3, std \ldots)$	$I_{np}(x, t_1, t_3, std \ldots)$
$I_p(x, t_1, t_2, decr \ldots)$			$I_{np}(x, t_1, t_3, decr \ldots)$

- **increasing/increasing**
  decreasing/decreasing
- **steady/increasing**
  steady/decreasing
- **steady/steady**

Where there are multiple possibilities, we will consider only the first, the remainder following from symmetry. If there is no entry, or if there is an entry and the criteria are not satisfied, then we enter the delay period to give any discernible trend time to emerge as discussed above.

We need to specify a value for the length of the delay period—this is defined by the delay factor parameter, *df*.

Whenever merging takes place, values for α_{np}, β_{np}, λ_{np} and μ_{np} are calculated as described in Section 2.4.

3.1. Increasing/increasing ⇒ increasing

The criteria are exactly the same as for merging two intervals historically; see Section 2.4.1.

The case of the criteria being satisfied, and the current interval being incorporated into the previous one is illustrated in figure 9(a). The case where this does not happen, and a delay period created, is illustrated in figure 9(b).

3.2. Steady/increasing ⇒ steady

Given a steady interval followed by an increasing interval one can infer a super-interval *steady* if the minimum and maximum values of the increasing interval (α_i and β_i) both lie between the minimum and maximum values of the steady interval (α_j and β_j).

3.3. Steady/steady ⇒ steady

Given two meeting steady intervals, if one of them is a simple interval then criterion 2 in Section 2 will always be satisfied, and the two intervals will always be merged.

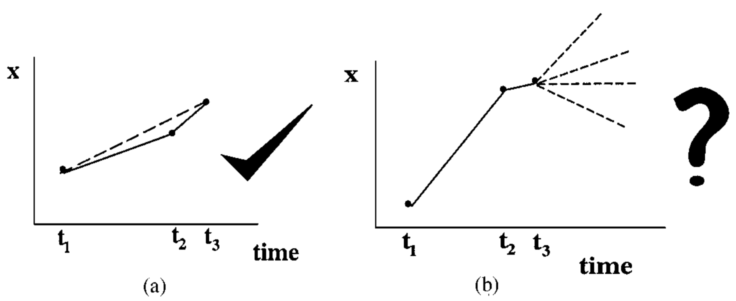


Figure 9. Inferring over two meeting *increasing* intervals.

4. Results

We will present the results of identifying temporal intervals in historical and real-time data.

4.1. Historical results

The above algorithm can be applied to data sets from any domain. We will study the types of temporal intervals generated by different values we use for the parameters *diff*, g_1, g_2 and *dur*.

Our results have been evaluated by a clinician.

In order to analyse the effect of our algorithm on the settings of the parameters an interesting case was chosen which contains many trends and events. The data set is a Central Venous Pressure (CVP) trace. The frequency of the data was one value every minute—this represents the average of the last sixty seconds of data every minute. The data was collected over a 24 h period and it contains 1440 data points. A median filter of size $k = 10$ was chosen. The original data set is shown in figure 10(a). Figures 10(b)–(e) are the graphical results of the intervals generated by different settings of the parameters.

Setting a high value for the parameter *diff* results in a small number of intervals being generated—these intervals are predominately long steady intervals. Setting *diff* to 5 results in only 23 intervals being generated—see figure 10(b). Only increasing and decreasing trends which have moderate and rapid rates are identified whereas slow changes are incorporated as part of a steady interval. The clinician was satisfied with these trends even though some increasing and decreasing trends were not identified. For example, although an increasing trend which began at approximately 04:00 was not explicitly identified, it can be derived by the new, clinically significant, raised steady interval which was identified instead.

Setting a high value for the parameters g_1 and g_2 results in a small number of intervals being generated. Setting g_1 to 25 and g_2 to 50 results in 36 intervals being generated (see figure 10(c)). Here no regard is given to different rates of change in the data. For example, in the original data set there are moderate increasing and decreasing intervals before and after the clinically insignificant event between 19:00 and 20:00—these are merged with the rapid increasing and decreasing intervals of the event. The clinician was again satisfied with these trends—he believes the derived trends are a true reflection of the data.

Setting a low value for the parameter *dur* results in many steady intervals being merged into increasing and decreasing trends. Setting *dur* to 0.05 results in 29 intervals being generated (see figure 10(d)). Here longer than expected increasing and decreasing trends are produced. Such trends can be considered as long term trends. The clinician was satisfied with these trends except the decreasing trend which began at approximately 06:00—he believes there was a more rapid and shorter decreasing trend at this time.

The best combination for this data set is to set the parameters *diff* to 1, g_1 to 3, g_2 to 6 and *dur* to 0.25 (see figure 10(e)). In this case, the clinician believes that all trends in the data were captured.

A summary of the effects of these setting are given in Table 4.

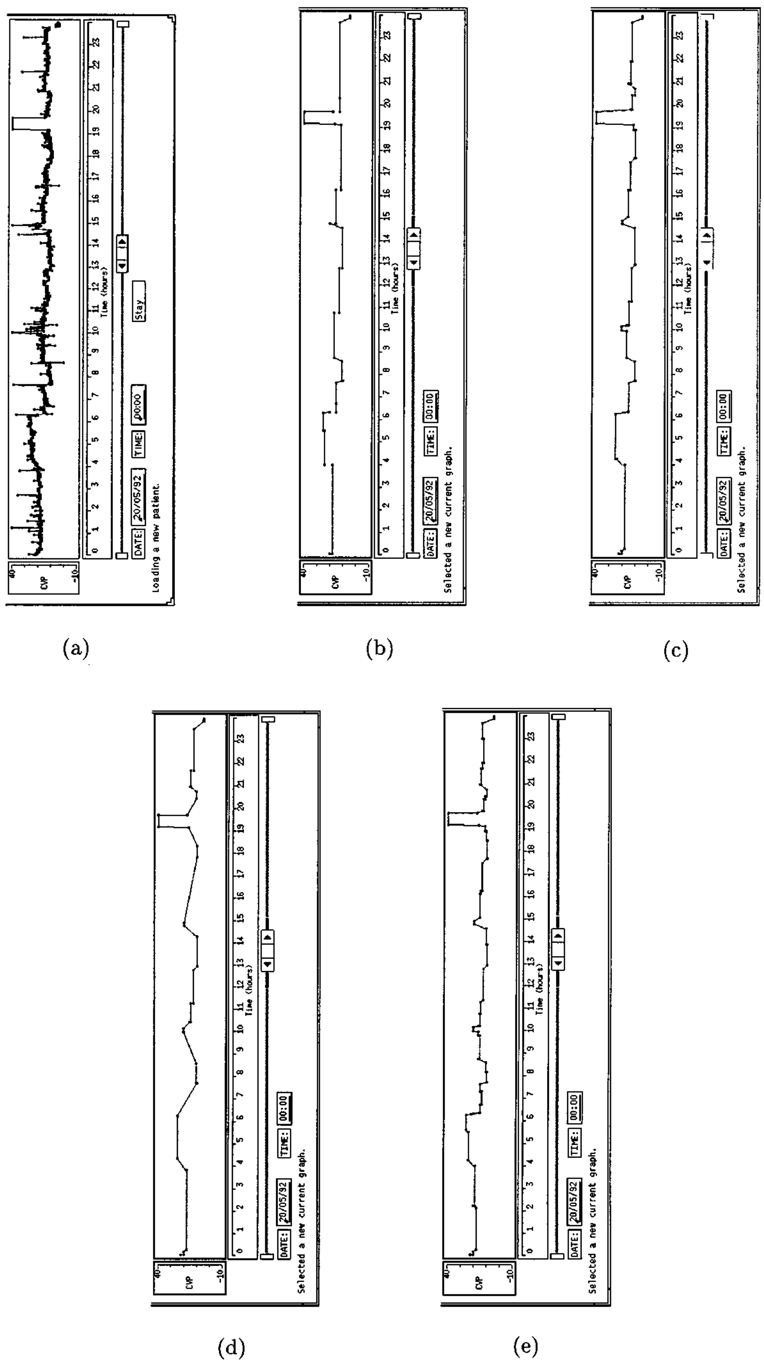


Figure 10. Results of using different parameter values to derive trends in historical data: (a) 1440 points; (b) 23 super temporal intervals; (c) 36 super temporal intervals; (d) 29 super temporal intervals; (e) 57 super temporal intervals.

Table 4. Results for various temporal inferencing parameters.

diff	g_1	g_2	*dur*	*Intervals*	Figure
High, e.g., 5	3	6	0.25	23	10(b)
2	High, e.g., 25	50	0.25	36	10(c)
2	3	6	Low, e.g., 0.05	29	10(d)
1	3	6	0.25	57	10(e)

The settings of the size of the filter window (k) for the *filter data* process and the parameters *diff*, *dur*, g_0 and g_1 for the *interval identification* process for historical data will depend on what data is being abstracted and in what context, the frequency of the data and how few or many intervals are desired.

4.2. Real-time results

We will analyse the results of our algorithm by using an interesting case which contains many trends and will present snapshots at times when there is a change in the current trend or when there is a delay. Note in our snapshots the left graph shows the original signal and the right graph shows the signal after a median filter has been applied to it at its leading edge. The abstraction displayed in the graphs is the current trend.

The data set is heart rate and its frequency is one value every minute. Appropriate values for the parameters were selected. A median filter of size $k = 5$ was chosen and we set the parameters for identifying trends in the heart rate to *diff* to 3, g_1 to 3, g_2 to 6, *dur* to 0.25 and *df* to 4.

In graph 11(a) it can be seen that the current trend for the heart rate at time 03:59:00 on 10/01/95 is steady.

In graph 11(b) it can be seen that at time 04:01:00 on 10/01/95 the current interval for the heart rate is *decreasing*. We now enter a delay period because we do not know if this new current interval represents the start of a new trend or a continuation of the current steady trend. Though we are in a delay, it is still assumed that the current trend is *steady*.

In graph 11(c) it can be seen that at time 04:05:00 on 10/01/95 the current trend for the heart rate has remained as *steady* i.e., we have inferred that within the delay period there was not a *decreasing* trend evolving. The *decreasing* interval encountered four minutes earlier was a momentary aberration. We are no longer in a delay.

In graph 11(d) it can be seen that at time 04:07:00 on 10/01/95 the current interval for the heart rate is *decreasing*. We now enter a delay period because we do not know if this new current interval represents the start of a new trend or a continuation of the current *decreasing* trend. Though we are in a delay, it is still assumed that the current trend is *steady*.

In graph 11(e) it can be seen that at time 04:09:00 on 10/01/95 the current trend for the heart rate has changed from being *steady* to *decreasing*. We inferred that within the delay period there was a *decreasing* trend evolving i.e., the *decreasing* interval encountered two minutes earlier was not a momentary aberration after all. We are no longer in a delay.

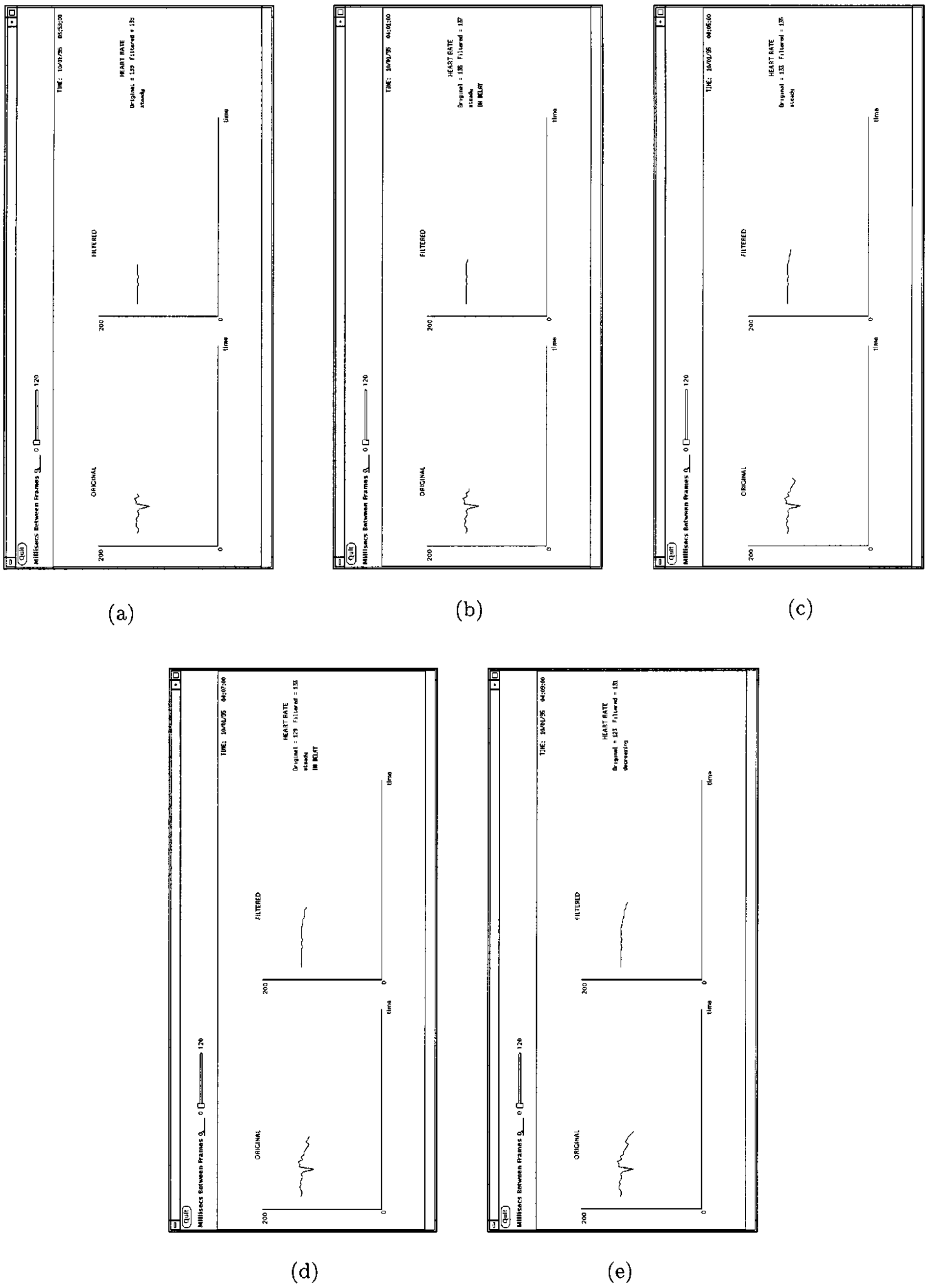


Figure 11. Results of deriving trends in real-time: (a) in a steady trend; (b) in a steady trend and in a delay; (c) in a steady trend; (d) in a steady trend and in a delay; (e) in a decreasing trend.

Using 03:59:00 on 10/01/95 as a starting point, we can say that the heart rate was *steady* from 03:59:00 on 10/01/95 to 04:09:00 on 10/01/95, and *decreasing* thereafter.

5. Related work

There are several systems which have dealt with time-series abstraction ((DeCoste, 1991), (Kahn et al., 1991) and (Russ, 1990)). In this section we will analyse and compare those systems which are most related to our work.

Our approach to the generation of interval abstractions is based on the work of Shahar (Shahar, 1997) who sets out a framework for generating abstractions of temporal data sets. In particular he discusses how two meeting (Allen, 1984) intervals can be combined as the result of executing a subtask which he calls temporal horizontal inference. Our rules for merging two intervals can be seen as an implementation of his horizontal inference table for gradient abstractions (Shahar, 1997, p. 104). Our rules for merging three intervals can be viewed either as an extension of the horizontal inference table to three intervals, or as a form of secondary gradient interpolation, where the middle interval constitutes (in Shahar's terms) a time gap.

Shahar's secondary gradient interpolation methods are similar to our system's methods for deriving trends and their gradients. Shahar uses C_π, a function of parameter π, that is defined either by the domain expert or through analysis of the distribution of π or different contexts. Using the C_π property minor absolute changes in the value of π that are less than a certain threshold can be ignored in order to identify general qualitative trends—this is similar to our *diff*, g_1 and g_2 parameters.

Shahar merges three meeting intervals into a single interval e.g., if parameter π can be interpolated by primary interpolation as, for example, *decreasing* over a gap interval I_g between two abstractions I_1 and I_2, then a *decreasing* abstraction can be inferred of π over I_g. In general, an overall *increasing* or *decreasing* abstraction might be created even when the gap interval could be abstracted as *same* (steady)—this is similar to using our *dur* parameter for merging three meeting intervals.

As far as we are aware, Shahar's knowledge-based temporal abstraction theory has mostly been applied to domains where the data is somewhat sparse (e.g., diabetes, cancer therapy, etc.); our work shows that it can be applied to much denser datasets where temporal horizontal inference is applied to reduce the number of intervals by factors of 50 or more. In addition we have demonstrated how the technique can be extended to the generation of interval abstractions in real time.

Our work is also similar to that of Larizza et al. (1995) who merge intervals to construct simple temporal abstractions. They too define context dependent parameters to control the merging, such as a minimal speed of change and a minimal duration for a super-interval. However their approach is designed to be applied to sparser datasets than ours—originally for monitoring protocols for recognising infection in heart transplant recipients and subsequently for diabetes. In such domains data may not be acquired on a periodic basis and they (like Shahar) are obliged to define a maximal distance between samples beyond which merging is not allowed. It is not clear to what extent merging is considered for more than two intervals at once—in the way that our approach considers three interval merges.

They state that their ‘algorithms for detecting increase, decrease and stationary abstractions are based on linear regression methods’ but there is no indication as to how the data subset for regression is selected beyond a statement that linear regression is applied to ‘different length sub-intervals’.

In the TrenDx System (Haimowitz and Kohane, 1996; Haimowitz et al., 1995) temporal abstraction into intervals is carried out as a necessary precursor to pattern matching to trend templates (an archetypal pattern of data variation for specific medical situation which is to be identified). TrenDx uses linear regression to fit gradients to the data; again the question arises as to how to select the data sub-set over which to perform the regression. TrenDx approaches this from two perspectives. Firstly, they assume that it will be possible to identify specific anchor points which will fix the start (or possibly the end) of the sub-set. In the only example which considers dense data sets (Haimowitz and Kohane, 1996) this is the onset of ‘handbagging’ (the delivery of 100% oxygen to an ICU patient on an intermittent basis by a nurse squeezing a bag). This event can be recognised from the setting of a switch on the ventilator. Secondly, TrenDx generates a tree of alternative chronologies; in terms of temporal gradient abstraction this means applying linear regression to a number of different sub-sets in parallel. Chronologies are then scored and pruned. Our approach differs in that (i) it makes no prior assumptions about the start/end points of our abstracted intervals—they emerge from the iterative temporal inferencing and (ii) it is deterministic—no scoring is used. TrenDx was initially developed in a domain (pediatric growth monitoring) in which the data frequency is somewhat low. It has been applied to the analysis of ICU data, but only to one patient; this makes it difficult to know how robust it will be when applied to large data sets.

High volume data from the neonatal ICU is handled by the VIE-VENT system (Miksch et al., 1996; Horn et al., 1997). Trends are measured in real time using linear regression, but in this case a fixed length window is used to define the data sub-set. More exactly, four windows are used to measure trends over the very short term (1 min), short term (10 min), medium term (30 min) and long term (3 h). These (and other) abstractions are performed on every incoming sample; there is no attempt to merge these abstractions into intervals and to reason about temporal relationships between intervals.

The ICU is also the source of high volume data for Dojat and co-workers (Dojat and Sayettat, 1996; Chittaro and Dojat, 1997). However their mechanism for generating higher level abstractions by aggregating (or merging) intervals only applies to parameter states and not to gradients (or trends). The desirability of extending their approach to trend abstractions is recognised (Chittaro and Dojat, 1997, p. 448).

DIA-MON-1 (Steimann and Adlassnig, 1994) is a system which classifies data streams into intervals through a set of constraints. Here the concept of trend is designed to model imprecise notions of courses and its trend detection is based on fuzzy classification. A series of time stamped data values belong to a fuzzy trend if there is a degree of match to the trend. However DIA-MON-1 can only abstract data to predefined trends and is based on moving windows each with its own fixed length; interval merging does not take place.

However, some of the above approaches are not applicable for the purposes of generating temporal intervals from dense data sets. Most of them assume that the data contains no noise. Though some of the above authors do handle noise, others do not. In many domains, noise is represented as the occurrence of events e.g., a faults in the measuring sensor. Data needs to be filtered to get rid of events otherwise unnecessary intervals representing these events will be generated. Incorporating noise does not reflect the true state of the system. Events need to be removed either by a standard filter or by the identification of intervals which represent an event. Removal of noise is domain dependent. Identified events need to be stored in an audit database for reference purposes.

Analysis of the validity of monitor data and applying repair and adjustment methods for correcting erroneous or ambiguous data is one way to deal with noise. (Horn et al., 1997) propose thirteen different methods for data validation and repair. These methods are categorised by their underlying temporal ontology (time-point, time-interval, or trend). Our system removes noise using a median filter. Any further non-physiological events can be removed by matching meeting intervals in one or, simultaneously, more variables (Salatian, 1997).

Post processing of noise-free data (e.g., applying a gaussian filter to smooth the data (Hau, 1994) is not applicable for deriving temporal intervals based on rate—rates are lost by the smoothing process (step like changes are deformed into a slope). Smoothing is acceptable if one is interested in only the sign of the derivative but is not applicable when one is interested in different rates of change of data. For dense data sets, different rates of change need to be captured—these can based on the gradient of the slope. Three kinds of rates can be defined—slow, moderate and fast.

From a real-time point of view, (Nelson and Hadden, 1994) uses an approach called State-Based Feature Recognition (SBFR) to recognise trends automatically in real-time. SBFR is *feature-based* in that each of its state machines operates by recognising patterns in its input data that it is designed to look at as a single item, a *feature*. Features to be recognised using SBFR are represented as finite state machines. The state machines are made up of a set of states $S_1, \ldots, S_n$ and transitions between those states, $T_1, \ldots, T_m$. Each state in the machine represents a stage in the identification of the feature. Each state has associated with it a set of transitions which may have one or actions associated with it to be performed whenever that transition is taken.

Each transition has a condition which defines when the transition may be traversed. When the machine's current state is the transition's *from* state and the condition for a particular transition is true then the machine traverses the transition.

The general meaning of a feature is captured by the machine's state and transitions, while the specifics of the feature (i.e., the exact data which causes the state machine to move from one state to another) are captured by the conditions.

Machines to recognise trends are called *trend machines*. To recognise, for example, a simple increasing trend, an increase can be defined as being two jumps in the data occurring greater than 5 time units apart, or three jumps in the data if the first two jumps happen in less than or equal to 5 time units. The reason for the time limit is to differentiate between an increase and a spike. Any time a decrease in the data is seen, the machine will transition back to the initial state. When the machine enters the *increase* state an increasing trend has been identified.

Our system derives trends in real-time by extending the merging algorithm used to identify temporal intervals in historical data.

We need a way to evaluate our results i.e., we need a formalism which can tell us if we have obtained the correct level of temporal abstraction. Aliferis et al. (1997) address the problem of providing and evaluating appropriate levels of temporal abstraction through a common formalism for medical decision-support systems. They compared querying a detailed model with an abstracted model of a medical system. The detailed model can answer an astronomical number of queries (most of which are of no clinical interest), whereas the abstracted model can only answer only a few queries (possibly excluding many queries of clinical importance). They conclude that as temporal abstraction increases, correctness in general will decrease. Similarly, our analysis of our results show that clinically significant short term changes in the data may not be identified when we are looking for long term trends—this, in turn, can affect the interpretation of the derived trends.

6. Summary and conclusions

We have presented a report which describes a way to derive trends in historical and real-time data. Our algorithm involves 2 consecutive processes: *filtering* and *interval identification*.

A filter is required which can remove events and retain the baseline signal of a parameter. From studies of various filters, we concluded that a median filter serves this purpose. Median filtering is a non-linear signal processing method able to remove both noise and transients from the signal without distortion of its base line. The median filter has two distinct advantages: good preservation of edges in the signal and excellent attenuation of impulsive noise.

A *merging* algorithm has been developed for generating temporal intervals from dense data sets. These temporal intervals have the attributes *increasing*, *decreasing* or *steady*. *Increasing* and *decreasing* trends can be classified into *slow*, *moderate* or *rapid* depending on their rate of change.

The algorithm for deriving temporal intervals in historical data involves initially interpolating between each data point to create simple temporal intervals then, repeatedly merging intervals which share similar characteristics into larger intervals until no more similarities can be found. The merging algorithm is achieved using temporal inference rules—this is based on knowledge of the specific signals e.g., clinicians may classify an interval in the mean blood pressure as *steady* if the mean blood pressure does not change by more than 5 mm Hg.

In real-time we have to consider if the latest data indicates a change in the trend or a momentary aberration. Our system derives trends in real-time by extending the merging algorithm used to identify temporal intervals in historical data.

We have used the temporal abstraction mechanisms described in this paper in three ways (Salatian, 1997):

1. removal of clinically insignificant events—this is achieved by matching meeting intervals in one or, simultaneously, more variables.
2. identifying clinical conditions—this is achieved by matching the values and/or trends of intervals in one or, simultaneously, more variables.

3. determining the outcome of therapies—this is achieved by comparing the interval when the therapy was administered to an interval in the future when the outcome of the therapy is expected to be achieved.

Our system has been tested on 9 different data sets each containing 3 or 4 different signals. In general our system identifies intervals where the expert agrees that a clinically insignificant event or clinical condition is present.

Deriving trends in clinical data is a difficult problem and we believe our system is a step forward in the development of systems for the interpretation of ICU data.

Acknowledgments

We would like to thank Dr. Andy Ronald, consultant anaesthetist at Aberdeen Royal Infirmary, UK, for evaluating our results.

References

Aliferis, C.F., Cooper, G.F., Pollock, M.E., Buchanan, B.G., and Wagner, M.M. (1997). Representing and Developing Temporally Abstracted Knowledge as a Means Towards Facilitating Time Modeling in Medical Decision-Support Systems, *Computers and Biology in Medicine*, 27(5), 411–434.

Allen, J.F. (1984). Towards a General Theory of Action and Time, *Artificial Intelligence*, 23, 123–154.

Chittaro, L. and Dojat, M. (1997). Using a General Theory of Time and Change in Patient Monitoring: Experiment and Evaluation, *Computers and Biology in Medicine*, 27(5), 435–452.

DeCoste, D. (1991). Dynamic Across-Time Measurement Interpretation, *Artificial Intelligence*, 51, 273–341.

Dojat, M. and Sayettat, C. (1996). Realistic Model for Temporal Reasoning in Real-Time Patient Monitoring, *Applied Artificuial Intelligence*, 10, 121–143.

Gallagher, Jr. N.C. and Wise, G.L. (1981). A Theoretical Analysis of the Properties of Median Filters, *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-29(6), 1136–1141.

Haddad, R.A. and Parsons, T.W. (1991). Digital Signal Processing Theory, Applications, and Hardware. *Design of DSP Algorithms*. Computer Science Press, chap. 5, pp. 184–273.

Haimowitz, I. and Kohane, I. (1996). Managing Temporal Worlds for Medical Trend Diagnosis, *Artificial Intelligence in Medicine*, 8(3), 299–321.

Haimowitz, I., Le, P.P., and Kohane, I. (1995). Clinical Modelling Using Regression-Based Trend Templates, *Artificial Intelligence in Medicine*, 7(6), 473–496.

Hau, D.T.-W. (1994). Learning Qualitative Models from Physiological Signals, Master of Science in Electrical Engineering and Computer Science, Massachusetts Institute of Technology.

Horn, W., Mikcsh, S., Egghart, G., and Popow, C. (1997). Effective Data Validation of High-Frequency Data: Time-Point-, Time-Interval-, and Trend-Based Methods, *Computers and Biology in Medicine*, 27(5), 389–409.

Kahn, M.G., Fagan, L.M., and Sheiner, L.B. (1991). Combining Physiological Models and Symbolic Methods to Interpret Time-Varying Patient Data, *Methods of Information in Medicine*, 30, 167–178.

Kohane, I.S. and Haimowitz, I.J. (1993). Hypothesis-Driven data abstraction with trend templates, *Symposium on Computer Applications in Medical Care*.

Larizza, C., Bernuzzi, G., and Stefanelli, M. (1995). A General Framework for Building Patient Monitoring Systems. *Proceedings of the 5th Conference of Artificial Intelligence in Medicine Europe* (AIME 95) (pp. 91–102). Pavia.

Miksch, S., Horn, W., Popow, C., and Paky, F. (1996). Utilizing Temporal Data Abstraction for Data Validation and Therapy Planning for Artificially Ventilated Newborn Infants, *Artificial Intelligence in Medicine*, 8(6), 543–576.

Nelson, K.S. and Hadden, G.D. (1994). Real-Time Feature Recognition in Medical Data, *Working Notes of AAAI Spring Symposium Series: Interpreting Clinical Data* (pp. 101–105). Stanford University.

Russ, T.A. (1990). Using Hindsight in Medical Decision Making, *Computer Methods and Programs in Biomedicine*, Elsevier, vol. 32, pp. 81–90.

Salatian, A. (1997). ASSOCIATE: The Interpretation of ICU Data Using ASSOCIAtional and TEmporal Knowledge, PhD thesis, Aberdeen University.

Shahar, Y. (1997). A Framework for Knowledge-Based Temporal Abstraction, *Artificial Intelligence*, 90(1/2), 79–133.

Steimann, F. and Adlassnig, K.-P. (1994). Two-Stage Interpretation of ICU Data Based on Fuzzy Sets, *Working Notes of AAAI Spring Symposium Series: Interpreting Clinical Data* (pp. 152–156). Stanford University.