The University of Nottingham

UNITED KINGDOM · CHINA · MALAYSIA

Prince, Rawle and Byrne, Matthew and Parry, Tony (2016) Meta-analytic framework for efficiently identifying progression groups in highway condition analysis. Journal of Computing in Civil Engineering, 30 (3). 04015044. ISSN 1943-5487

A Meta–Analytic Framework for Efficiently Identifying Progression Groups in Highway Condition Analysis

Rawle Prince [1], Matthew Byrne [2], Tony Parry [3]

ABSTRACT

The **MML2DS** (Minimum Message Length Two Dimensional Segmenter) criterion is a powerful technique for road condition data analysis developed at the Nottingham Transportation Engineering Centre (NTEC), University of Nottingham. The criterion analyses condition data sets by simultaneously identifying optimum trends in condition progression, the position in time and space of maintenance interventions, longitudinal segments within links, and the error likelihood of each measurement. This is done in an unsupervised manner via classification and regression models based on the Minimum Message Length metric (**MML**). Use of MML, however, often requires an exhaustive comparison of all possible models, which naturally raises considerable search–control issues. This is precisely the case with the **MML2DS** approach. This paper presents an efficient meta–analytic framework for controlling the generation of *progression groups*, which considerably reduces the search space prior to the application of **MML2DS**. This is achieved by identifying 'founder sets' of longitudinal segments, around which families of segments are likely to be formed. An effective subset of these families is then selected, after which the **MML2DS** criterion is employed as the final arbiter to determine ultimate model configurations and fits. This approach has proved to be very powerful, resulting in significant improvements in efficiency to the effect that accurate results are obtained in a few minutes where it previously took weeks with much

[1]Yotta, Yotta House, 8 Hamilton Terrace, Leamington Spa, CV32 4LY, Warwickshire. Email: rawle.prince@gmail.com

[2]And/Orr Limited, 14 Clarendon Street, Nottingham, NG1 5HQ, United Kingdom. Email: drmattbyrne@gmail.com

[3]Nottingham Transport Engineering Centre, Faculty of Engineering, University of Nottingham, Nottingham, NG7 2RD, United Kingdom. Email: tony.parry@nottingham.ac.uk

smaller data sets. The indications are that this approach can be applied to other techniques besides **MML2DS**.

INTRODUCTION

Road agencies collect expansive data sets of pavement condition, forming the backbone of the asset management systems, which are used to identify various performance indicators and maintenance needs. Very often, the data collected is used to fit time series — termed progression rates — in order to better understand surface condition indicators, such as pavement roughness and rutting. A road network under study may have many thousands of kilometres of pavement, typically divided into a series of sections: $\mathcal{N} = \{\mathcal{S}_i | i \in \{1, 2, \ldots, m\}\}$. Each section $\mathcal{S}$ is subsequently subdivided into a series of discrete–length[1] chains $\mathcal{S}_i = \{\mathcal{C}_{i1}, \ldots, \mathcal{C}_{in}\}$, where $\mathcal{C}_{ij}$ denotes chain j of section i, and data for individual chains would be recorded over a number of measurement intervals, usually years. For instance, a typical chain $\mathcal{C}_j = \{x_1, \ldots, x_p\}$ would comprise a series of measurements x_j, recorded at various measurement periods, over a number of years. **Table. 1** gives an example of simulated rutting data for a 1800 meter road segment over an eleven year period. The measurements are often subject to noise or errors which, together with issues of unrecorded maintenance, changes in the measurement devices, as well as possible seasonal variation can combine to make the task of estimating current condition, or identifying true progression rates, very difficult.

The **MML2DS** criterion introduced in (Byrne and Parry 2009) has proved to be very effective in identifying true trends in condition progression, the position in time and space of maintenance interventions, longitudinal segments within links, and the severity of errors among measurements. The key idea was to share data among adjacent chains in a section in order to identify *progression groups*, $\mathcal{G}_i^{\mathcal{S}}$ for a section $\mathcal{S}$, formed from chains that can be described by a common progression rate and associated maintenance intervention pattern:

$$\mathcal{S} = \bigcup_{\mathcal{G} \in \mathcal{G}_i^{\mathcal{S}}} \mathcal{G}, \text{ where } \mathcal{G} = \{\mathcal{C}_1, \ldots, \mathcal{C}_k\}.$$

[1]Typically 10 meters, averaged over 100 meters.

Ultimately, the criterion also identifies how data in individual measurements within a group relate to the group's progression rate and maintenance intervention pattern, giving valuable information in terms of possible measurement errors and/or seasonal variation. **Fig. 1** shows a progression group model for the data in **Table. 1**.

Progression group models in (Byrne and Parry 2009) were identified using Minimum Message Length (**MML**) inference (Wallace 2005). **MML** is a powerful technique for inductive inference, residing at the intersection of Information Theory and Statistics, which seeks to minimise an objective function that estimates the validity of an inferred model. Since it was first introduced (Wallace and Boulton 1968), **MML** has been successfully applied to numerous settings, often outperforming rival techniques. These include, selecting the configuration of Neural Networks (Makalic et al. 2009), classification of proteins in DNA (Zakis et al. 1994), grouping ordered data (Fitzgibbon et al. 2000), inferring decision graphs (Tan and Dowe 2003), classification of spatial data (Wallace 1998), clustering of protein structures (Edgoose et al. 1998) and bushfire prediction using decision trees (Dowe and Krusel 1993). The issue with **MML**, however, is that one can only be certain that the optimum model has been identified after the metric has been applied to all other models. This is very much the case with the **MML2DS** criterion, especially with regard to the identification of progression groups. Considering all possible models is not an issue when dealing with small sections. However, there is an exponential increase in the number of possible progression group models that can be obtained from a given section, and checking all of them quickly becomes problematic as section lengths increase. Moreover, real world pavement networks can have sections with hundreds or thousands of chains and testing all progression group models in such settings is intractable.

This paper presents a meta–analytic framework for pre–processing progression group models in order to mitigate search control issues that arose during the application of the **MML2DS** criterion. Rather than checking all possible progression group models generated from a section with the **MML2DS** criterion, a relationship metric is employed as a heuristic to define initial groups around which progression groups are likely to be formed. These initial groups subsequently form the nucleus of larger groups, which are subsequently evaluated by a fitness function derived from the relationship metric. The 'fittest' progression group models are retained, and it is these that are ultimately analysed by the **MML2DS** criterion. More often that not, the set of progression group models retained is not only significantly smaller than the set of possible progression group models obtainable from a given section, but also contains the desired model. Hence, checking this reduced set with the **MML2DS** criterion generally leads to a result considerably faster that would otherwise be the case.

This approach can be thought of as a form of subspace clustering (Vidal 2011), and is comparable to heuristic techniques typically used to deal with combinatorial explosion in this setting (Aggarwal et al. 1999; Kriegel et al. 2005). The speed–ups in the analyses were considerable, especially when it came to large sections, returning results in a few minutes where it previously took weeks, whilst maintaining the required level of accuracy.

The paper is organized as follows. The next section provides a detailed presentation of the meta–analytic framework together with algorithms for its implementation. The section that follow discusses results and outputs obtained from experiments, while concluding remarks are in the section thereafter.

THE META–ANALYTIC FRAMEWORK

Suppose a section with n chains $\mathcal{S} = \{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$ is given, where the aim is to determine the number of progression group models that can be generated for $\mathcal{S}$. The number of chains in a progression group can be set to a minimum k, and let m be the number of progression groups that can be obtained from $\mathcal{S}$. The number of possible progression group models obtainable from $\mathcal{S}$, each with m progression groups, can be given by:

$$\Phi(m, n) = \begin{cases} 1 & \text{if } m = 0 \\ \\ \sum\limits_{i=k}^{n-k} \Phi(m-1, n-i) & \text{otherwise.} \end{cases} \tag{1}$$

Consequently, the number of possible ways of combining at least m chains is given by $\Omega(m,n)$:

$$models(m,n) = \sum_{m=0}^{n/k} \Phi(m,n), \tag{2}$$

where x/z denotes the integer quotient of x by z. **Fig. 2** shows how the number of possible progression group models increases for values of n with $k = 1$. As can be seen, setting $n = 15$ yields 16383 possibilities, and increasing n to 21 and 23 yields 1048575 and 4194303 possibilities, respectively. This is approximately $O(1.935)^n$, so setting $n = 60$ yields a value well over one billion. Generating all of these possibilities on its own can be computationally expensive, and application of the **MML2DS** criterion to a 5 kilometre section, for instance, using the original approach is clearly not feasible.

The Main Idea

The technique presented is based on the idea that progression groups are formed around core members, or *founder sets*, to which other members are subsequently allocated. A relationship metric is employed to discover initial founder sets, which are subsequently re–combined to form a preliminary set of progression group models. Members of this preliminary set are then tested using a sort of fitness function obtained by estimating the strength of the stated relationship among members of a progression group, averaged over all progression groups in a model, and are selected or discarded based on how they compare to previously tested progression group models. It is this reduced set of progression group models, with closely related members, that is submitted to **MML2DS** criterion for final analysis. The algorithm is shown in **Fig. 3**.

As shown in **Fig. 3**, given a section $\mathcal{S}$ the founder sets $\mathcal{S}^x = \{\mathcal{X}_1, \mathcal{X}_2 \ldots \mathcal{X}_n\}$ for $\mathcal{S}$ are first calculated, where each $\mathcal{X}_i = \{\mathcal{C}_{i1}, \ldots, \mathcal{C}_{in}\}$ is a close set of chains subject to a stated meta–relationship and tolerance, such that $\mathcal{S} = \bigcup_{\mathcal{X} \in \mathcal{S}^x} \mathcal{X}$. Let $\mathcal{N} = \{\mathcal{S}_i | i \in \{1, 2, \ldots, m\}\}$ be a network under study. $\mathcal{R} \in \mathcal{C} \times \mathcal{C} \to \mathbb{R}$ is a meta–relationship for $\mathcal{N}$ if there is a least upper bound on $\mathcal{R}$ — i.e. $\exists \tau. \forall \mathcal{S}_i \in \mathcal{N}, \forall x, y \in \mathcal{S}_i. \mathcal{R}(x, y) \leq \tau$. It is also important that $\mathcal{R}$ is defined such that τ denotes the strongest possible relationship under $\mathcal{R}$. A close set subject to a given meta–relationship is subsequently defined as follows.

Definition 1 (close set) *Let $\mathcal{X}$ be a set of chains in a section $\mathcal{S}$ and $\mathcal{R} \in \mathcal{C} \times \mathcal{C} \to \mathbb{R}$ the meta–relationship on the network containing $\mathcal{S}$. For a given tolerance η, where $\eta < \tau$, $\mathcal{X}$ is a η–close set of chains, subject to $\mathcal{R}$, if $\forall x, y \in \mathcal{X} . \mathcal{R}(x, y) \in [\eta, \tau]$.*

Since founder sets are intended to initiate progression groups, and not replace them, the relationship metric $\mathcal{R}$ should satisfy a necessary condition for the formation of progression groups. For instance, if $\forall x \in \mathcal{C}_i, y \in \mathcal{C}_j . x \neq y$, but $\mathcal{C}_i$ and $\mathcal{C}_j$ share the same mean and standard deviation, it would be very likely that $corr(\mathcal{C}_i, \mathcal{C}_j) \in [\eta, 1]$, where $corr$ denotes the Pearson correlation coefficient and η some value between 0 and 1 which specifies a high likelihood of closeness relative to the standard deviation — e.g. 0.85 for standard deviation 1.5. Once the founder sets have been identified, a set of progression group models $\mathbb{G} = \{\mathcal{G}_1^{\mathcal{S}^x} \ldots \mathcal{G}_n^{\mathcal{S}^x}\}$ is then generated from $\mathcal{S}^x$ by considering all re–combinations of $\mathcal{S}^x$ such that each $\mathcal{G}_i^{\mathcal{S}^x} = \{\mathcal{G}_{i1}, \ldots, \mathcal{G}_{iq}\}$, and $\mathcal{G}_{ik}$ is a union of founder sets.

Depending on the definition of $\mathcal{R}$ and the value of τ, the number of elements in $\mathbb{G}$ can be very large, so relying solely on the generation of founder sets can result in little improvement over employing the **MML2DS** criterion to all possible progression group models. The next step, therefore, is to build a smaller set of potential progression group models for analysis by the **MML2DS** criterion in such a way that the cardinality of the reduced set is likely to be considerably less than the number of possible progression group models that can be generated from $\mathcal{S}$. This is achieved by first defining the *connectedness* of a progression group, which is then averaged over all groups in a progression group model to estimate a 'fitness' score for the progression group model.

Definition 2 (connectedness) *For any progression group $\mathcal{G}$ with cardinality k, the con-*

nectedness of the chains in $\mathcal{G}$*, subject to* $\mathcal{R}$*, is given by*

$$con(\mathcal{G}) = \begin{cases} \lambda & \text{if } k < 2 \\ \sum_{i=1}^{k-1} \frac{g(\mathcal{G}[i], \mathcal{G}[i+1])}{k-1} & \text{otherwise} \end{cases} \tag{3}$$

where λ *is a default value for groups with less than* 2 *chains,* $\mathcal{G}[i]$ *is the* i^{th} *chain in* $\mathcal{G}$ *and* $g(a,b) = |\mathcal{R}(a,b) - \tau|$*, for* $a \neq b$ *and* a *adjacent to* b*.*

Note that since τ is the upper bound on $\mathcal{R}$ it follows that for a given progression group $\mathcal{G}$, the proximity of $con(\mathcal{G})$ to 0 is proportional to the strength of the relationships between adjacent chains in $\mathcal{G}$. Correspondingly, (4) provides a means of quantifying the strength of relationships within a progression group model $\mathcal{G}_i^{\mathcal{S}x}$ obtained from a section $\mathcal{S}$, based on the connectedness of progression groups within it.

$$con_M(\mathcal{G}_i^{\mathcal{S}x}) = \sum_{j=1}^{m} \frac{con(\mathcal{G}_{ij})}{m}, \tag{4}$$

where m is the cardinality of $\mathcal{G}_i^{\mathcal{S}x}$. Consequently, con_M can be thought of as a fitness function for progression group models, and is employed so that increasingly 'fitter' models will 'survive' in order to be examined by the **MML2DS** criterion.

Implementation

Although the technique was developed in the context of the **MML2DS** criterion, it is clearly applicable to settings where other metrics may be employed. It was therefore implemented as a generic, higher–order function which takes the following inputs:[2]

1. a generic list of elements to combine. In the context of the **MML2DS** criterion, this list is instantiated to a list of arrays denoting a section, where each array represents measurements over a finite number of years for a given chain in the section.

[2] An example implementation in C# is available online (Prince 2015), as well as a demonstration of the technique on the section data in **Table. 1**.

2. a function representing the relationship metric which takes as input a pair of values of the type contained in the input list, and returns a real number.[3]
3. a value for the upper bound (or denoting the strongest relation) of the relationship function.
4. a value for the tolerance η used to identify founder sets.
5. a specification of the comparison operation to be used when selecting progression group models for final analysis.

The function outputs a list containing lists of lists of elements from the input list. For instance, the output in the context of the **MML2DS** criterion is a list of progression group models, each of which is represented by a list of list of arrays.[4]

Notation The notation used in the algorithms below is as follows. Lists are denoted by square brackets, for example $[\mathbb{R}]$ is a list of real numbers and $[X]$ a list of any type X. $[]$ denotes an empty list or sequence, while subscripts are used to refer to elements in a list, for instance xs_2 refers to the second element of the list xs. *len* is a function that returns the length of a list. Given a value x and a list xs, $(x :: xs)$ is a list with x added to the front of xs, while $(x <> xs)$ is $(x :: xs)$ providing that x is not already at the front of xs:

$$(x <> xs) = \begin{cases} (x :: xs) & \text{if } xs = [] \vee xs_1 \neq x \\ xs & \text{otherwise.} \end{cases}$$

For a given list xs and some integer i, $xs(\leq i)$ and $xs(> i)$ denote the first i values of xs and the remaining values of xs, respectively. Finally, *maxLen* takes a list of lists as input and returns the length of longest element in the input list.

[3]This is represented as a function delegate in (Prince 2015) while a function pointer can be used in languages such as C or C++.

[4]The implementation in (Prince 2015) returns an additional value denoting the number of founder sets generated. This is included for evaluation and can be easily omitted if required.

Algorithm 2.1 Algorithm for identifying founder sets. The main function, *founders* is called with $acc = []$.

Function: $founders(ls, \mathcal{R}, \tau, ac)$
  if $ls = []$ **then**
    return acc
  else if $len(ls) = 1$ **then**
    $als \leftarrow (ls_1 :: acc)$
    return als
  else
    $efs \leftarrow gps(1, ls, ls_1, \mathcal{R}, \tau, [])$
    $m \leftarrow maxLen(efs)$
    $ft \leftarrow ls(\leq m)$
    $bk \leftarrow ls(> m)$
    $acf \leftarrow (ft :: ac)$
    return $founders(bk, \mathcal{R}, \tau, acf)$
  end if

Function: $gps(n, ls, e, \mathcal{R}, \tau, acc)$
Require: $n \geq 0 \wedge acc \neq []$
  if $(n > len(ls) - 1)$ **then**
    return acc
  else
    $xs \leftarrow ls(\leq n)$
    $valid \leftarrow \forall x \in xs.\, \mathcal{R}(e, x) \leq \tau$
    if *not valid* **then**
      return acc
    else
      $ys \leftarrow (xs :: acc)$
      return $gps(n + 1, ls, e, \mathcal{R}, \tau, ys)$
    end if
  end if

Identifying founder sets

The function to identify founder sets is shown in **Algorithm. 2.1**. It takes the input list (i.e. the representation of the section $\mathcal{S}$), the relationship metric $\mathcal{R}$, the tolerance τ and a list which serves as an accumulator. An auxiliary function *gps* is used to identify a block $\mathbf{B}_i$ of elements such that $\forall x \in \mathbf{B}_i.\, \mathcal{R}(a, x) \leq \tau$, where a is the first element in the list. Each $\mathbf{B}_i$ identified is a founder set, and is subsequently removed from the list and added to the accumulator. The function is then applied recursively to the remaining elements of the input list and the accumulated $\mathbf{B}_i s$ are returned when the input list is empty.

Re–combining founder sets

The algorithm used to recombine founder sets to form progression group models, shown in **Algorithm. 2.2**, is based on (2). The main function *allGroups* implements (2) with $k = 1$. It re–combines the founder sets by accumulating the group models with i groups that can be formed from a list xs, where $i = 1, 2, \ldots, len(xs)$, and where the group models with i elements that can be formed from xs are given by the function *ngroups*, which implements (1). To form a group model with n elements from a list xs, with each group within the model containing at least k elements, for every $j = k \ldots (len(xs) - k)$, *ngroups* makes a group with the first j elements of xs then recursively forms $n-1$ groups from the remaining $ls(>j)$. The subsidiary groups are then combined with previous ones to form a group model with j groups, and each group model is subsequently added to the accumulator.

Algorithm 2.2 Calculating the possible groups from a generic list ls. The main function, *allGroups* is called with $acc = []$.

Function: $allGroups(xs, acc)$
Require: $xs \neq []$
  $n \leftarrow len(xs)$
  for $i = 0$ to $(len(xs) - 1)$ **do**
    $ys \leftarrow ngroups(i, 1, xs, [])$
    for $j = 1$ to $len(ys) - 1$ **do**
      $acc \leftarrow (ys_j :: temp)$
    end for
  end for
  return acc

Function: $ngroups(n, k, ls, acc)$
Require: $k > 0 \wedge ls \neq []$
  if $n \leq 0$ **then**
    return $([ls] :: acc)$
  else
    for $i = k$ to $(len(ls) - k)$ **do**
      $ft \leftarrow ls(\leq i), \quad bk \leftarrow ls(> i)$
      $xs \leftarrow ngroups(n-1, k, bk, [])$
      for $j = 1$ to $len(xs)$ **do**
        $x \leftarrow xs_j, \quad zs \leftarrow (ft :: x)$
        if $len(zs) \geq k$ **then**
          $acc \leftarrow (zs :: acc)$
        end if
      end for
    end for
    return acc
  end if

Applying the fitness test

The list of progression group models returned by **Algorithm. 2.2** is then processed using the function *mtBy* below

$$mtBy(f, ls) = \begin{cases} [] & \text{if } ls = [] \\ mtByAux(f, xs_1, xs(>1), []) & \text{otherwise,} \end{cases}$$

where the function *mtByAux* is given in **Algorithm. 2.3**. As shown, *mtByAux* takes a generic list xs, a (fitness) function f to be applied to elements of xs, the first element a from xs, and an accumulator zs which serves as the queue in **Fig. 3**. Every subsequent element of ls is compared to a. If an element y is deemed to be 'fitter' than a, it is added to the queue and y is then considered as the 'fittest' element so far. Otherwise, it is bypassed and a is compared to the next element of the list. Comparison in done using the operator *compare* which specifies the comparison to use when short–listing progression group models to the queue. In accordance with the desired generality of the implementation, given values x and y, *compare* can be set to either: (i) $x < y$, (ii) $x \leq y$ and (iii) $|y - x| < \epsilon$ for some $\epsilon \in (0, 1)$. The last option generalises the others in that it allows a group to be added if its fitness score (4) is within a defined proximity of those previously added to the queue.

Algorithm 2.3 Maintaining the 'fittest' elements of a list subject to a fitness function f.

Function: $\mathit{mtByAux}(f, a, \mathit{xs}, \mathit{zs})$
  if $\mathit{ls} = []$ **then**
    return zs
  else
    $x \leftarrow \mathit{xs}_1, \quad n \leftarrow \mathit{len}(\mathit{xs})$
    $\mathit{ls} \leftarrow \mathit{xs}(> n - 1)$
    if *compare* $(f(a), f(x))$ **then**
      return $\mathit{mtByAux}(f, a, \mathit{ls}, (a <> \mathit{zs}))$
    else
      return $\mathit{mtByAux}(f, x, \mathit{ls}, (x <> \mathit{zs}))$
    end if
  end if

RESULTS AND VISUALISATIONS

The framework was evaluated, independently and together with the **MML2DS** criterion, on simulated data for a number of pavement sections with various lengths, and with predefined amounts of progression groups and intervention points. Data for each group within a section was randomly sampled from a normal distribution with a unique mean and standard deviation, relative to the other groups within that group.

In order to test the framework's ability to reduce the number of generated progression group models, it was applied to a number of sections without any subsequent analysis. The data in **Table. 1** was one of these sections. There are two predefined progression groups in this section giving rise to the following progression group model $\{\{\mathcal{C}_1, \ldots, \mathcal{C}_5\}, \{\mathcal{C}_6, \ldots, \mathcal{C}_{18}\}\}$ as shown in **Fig. 1**. Applying **Algorithm. 2.2** to this section returns 131071 possible progression groups. However, after letting $\mathcal{R}$ be the Pearson correlation coefficient, and setting $\eta = 0.75$, $\tau = 1$ and the comparison *compare* such that $compare(a, b) = |a - b| < 0.03$, the meta–analytic framework reduces this to 12 possibilities, amongst which *is* the expected progression group model.[5]

For all of the sections evaluated, applying the **MML2DS** criterion to all possible progression groups models would have taken days to complete,[6] in addition to possible space complexity issues due to the generation of progression group models for long sections. It was, therefore, not feasible to compare the time it took the implementation of the **MML2DS** criterion combined with the meta–analytic framework to one without the meta–analytic framework. Instead, we investigated the trade off between accuracy and efficiency provided by the meta–analytic framework, and so examined the number of founder sets identified, the number of progression groups discovered, and the time it took to complete the analysis. In this way, the aim was to determine if the chosen relationship, the number of founder sets obtained and the subsequent reduction in the time it took to complete the analysis, had any significant impact on the accuracy of the analysis. Results obtained using the Pearson correlation coefficient *corr* as the relationship $\mathcal{R}$ are shown in **Table. 2**.

As these results show, we were able to discover the expected number of progression groups on every occasion, even when the section lengths were very large. These results compare with what was obtained with the original implementation of the **MML2DS** criterion (Byrne and Parry 2009), but, in this case, results were obtained in less than fifteen minutes, even with the longest sections, where it took upwards of five days for sections with less than 60 chains in (Byrne and Parry 2009). While part of this increase in performance can be attributed to our use of parallel programming techniques to exploit multi–core architectures during interactions of piecewise and mixture models, the identification of founders sets, and the subsequent selection of progression groups based on connectedness, considerably reduced the number of cases to be checked by the **MML2DS** criterion, and was clearly the main

[5] Note, this example is implemented in (Prince 2015).

[6] The tests were done on a 64 bit Windows 7 machine with 8GB RAM and an Intel Core i7–4800, 2.7GHz processor.

reason for the performance improvements.

This also shows that the meta–analytic framework does provide an effective technique for balancing the trade–off between efficiency and accuracy during the application of **MML** analysis. Moreover, not only can the meta–relationship function be adapted to different settings, but the parameters, for controlling the relationship's strength a well as the search space, can also be tailored to performance requirements on different systems, or to different domains. This approach clearly goes a long way in addressing complexity issues related to the **MML2DS** criterion, since, as can be seen from **Table. 2**, the time taken for results to be obtained depends on meta–relationships within the data set — indicated by the number of founder sets discovered — and not necessarily the size of the data set.

A major limitation of this approach, however, is that it might not always be straightforward to identify a suitable meta–relationship. Our use of the Pearson correlation coefficient was justified since data in each of the predefined progression groups was sampled from the same normal distribution. In other domains, one would expect that a fair amount of domain knowledge and/or experimentation would be required before a suitable meta–relationship can be identified.

Visualisations

The primary purpose of the meta–analytic framework was to control the generation of progression groups prior to **MML2DS** analysis, so outputs obtained from the final system, which employed the **MML2DS** criterion, corresponded to those obtained in the original application of the **MML2DS** criterion (Byrne and Parry 2009). As mentioned earlier, the aim of the **MML2DS** criterion was to identify the progression rates of the condition data. Example results are presented as shown in **Fig. 5** and **Fig. 4**. The position of maintenance interventions and progression groups are shown in coloured blocks in **Fig. 5**, whereby each block is a group of adjacent intervals which share a common progression rate. Progression rates for selected intervals and measurement errors (i.e. outliers) are shown at the right, with the lower section describing the likelihood of each data point being erroneous, in relation to the progression trend above it. For example, section 230 – 240 has a clear maintenance intervention occurring between years 2004 and 2005.

Fig. 4 uses a colour coding to highlight where and when errors in the condition data appear to exist. A deeper shade of red or blue indicate a higher likelihood of erroneous data, where red indicates those above the trend and blue those below the trend. For instance, there is a clear disparity in the measurements data recorded chain 10 – 20 in 2001 and since this is inconsistent with measurements taken in the proceeding and following years, it is highlighted as an error and not caused by a maintenance intervention. This disparity may have been caused, for instance, by a poorly calibrated device which overestimated condition levels along the whole section that year. **Fig. 5** also displays the position of measurement errors relative to the progression trend, which is displayed in a similar way to **Fig. 4**.

CONCLUSION

This paper presented a meta–analytic framework for pre–processing group permutations generated during the application of the **MML2DS** criterion. While the **MML2DS** criterion provides a novel solution to the problem of identifying progression rates, the required sharing of data over adjacent chains raised considerable search control issues, which potentially limited its applicability to real–world settings.

By applying a relationship that satisfies a necessary condition for the formation of progression groups, and estimating the relative connectedness of progression groups based on this relationship, the proposed meta–analytic framework provides a robust method of reducing the number of progression group models submitted to the **MML2DS** criterion for analysis. Empirical test have shown that, depending on the relationship selected and the choice of associated parameters, the set of progression group models retained usually contain the desired solution. The meta–analytic framework, therefore, provides an efficient and effective approach to managing the trade off between efficiency and accuracy required for applications of the **MML2DS** criterion, and **MML** in general, to real–world settings. There is no limitation to the meta–relationship that can be used, which clearly lends itself to the application of different techniques, for example fuzzy logic. Moreover, the framework was implemented as a generic function and can be utilised in different settings, and with various relationship functions. However, some understanding of the data set and the problem domain would be required to make effective use of this approach.

The framework also illustrates how novel search control techniques and quality data mining algorithms can be combined to extract information from noisy data sets without any significant loss in accuracy. While the progression rates were the ultimate answer sought by the **MML2DS** criterion, the progression groups obtained can provide useful information about past maintenance interventions. This would certainly be desirable in situations where maintenance records are not up–to–date, and knowledge of past maintenance can be used to derive strategies for the future. The next step is to apply this combined technique to real–world data, and we are in the process of doing so at present.

ACKNOWLEDGEMENT

This work was completed when the first and second authors were a research assistant and fellow, respectively, in the Nottingham Transport Engineering Centre, Faculty of Engineering, University of Nottingham. The research was supported by the EPSRC grant EPSRC/TSP 15TP with additional support from Devon County Council.

APPENDIX I. REFERENCES

Aggarwal, C., Wolf, J., Yu, P., Procopiuc, C., and Park, J. S. (1999). "Fast algorithms for projected clustering." *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA, ACM, 61–72.

Byrne, M. and Parry, T. (2009). "Network level pavement condition preparation using Minimum Message Length." *Twelfth International Conference on Structural and Environmental Engineering Computing*.

Dowe, D. L. and Krusel, N. (1993). "A decision tree model of bushfire activity." *Proceedings of the 6th Australian Joint Conference on Artificial Intelligence*, World Scientific , Singapore.

Edgoose, T., Allison, L., and Dowe, D. L. (1998). "An MML classification of protein structure that knows about angles and sequence." *Proceedings of the PACIFIC SYMPOSIUM ON BIOCOMPUTING*, World Scientific , Singapore.

Fitzgibbon, L. J., Allison, L., and Dowe, D. L. (2000). "Minimum Message Length grouping of ordered data." *Algorithmic Learning Theory*, H. Arimura, S. Jain, and A. Sharma, eds., Vol. 1968 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 56–70.

Kriegel, H., Kröger, P., Renz, M., and Wurst, S. (2005). "A generic framework for efficient subspace clustering of high–dimensional data." *Proceedings of the Fifth IEEE International Conference on Data Mining*, IEEE Computer Society, 250–257.

Makalic, E., Allison, L., and Dowe, D. L. (2009). "MML inference of single–layer neural networks." *Proceedings of the IASTED International Conference on Artificial Intelligence and Applications, 08 September 2003 to 10 September 2003*.

Prince, R. (2015). "C# implementation on the Meta–Analytic Framework. `https://github.com/rawlep/MetaAnalyticFramework/tree/ASCE`.

Tan, P. and Dowe, D. L. (2003). "MML inference of decision graphs with multi–way joins and dynamic attributes." *AI 2003: Advances in Artificial Intelligence*, T. Gedeon and L. Fung, eds., Vol. 2903 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 269–281.

Vidal, R. (2011). “Subspace clustering.” *Signal Processing Magazine*, 28(2).

Wallace, C. S. (1998). “Intrinsic classification of spatially correlated data.” *The Computer Journal*, 41(8), 602–611.

Wallace, C. S. (2005). *Statistical and Inductive Inference by Minimum Message Length*. Springer-Verlag (Information Science and Statistics).

Wallace, C. S. and Boulton, D. M. (1968). “An Information Measure for Classification.” *The Computer Journal*, 11(2), 185–194.

Zakis, J., Cosic, I., and Dowe, D. (1994). “Classification of protein spectra derived for the Resonant Resonant Recognition model using the Minimum Message Length principle.” *Proceedings of the 17th Australian Computer Science Conference (ACSC-17)*.

List of Tables

Chains	**Rutting Values**										
1	3.199	3.241	3.33	3.383	3.439	3.518	3.56	3.601	3.708	3.705	3.786
2	3.223	3.246	3.321	3.406	3 .451	3.514	3.555	3.639	3.725	3.781	3.857
3	3.204	3.236	3.291	3.387	3.474	3.53	3.602	3.682	3.752	3.834	3.875
4	3.167	3.247	3.346	3.444	3.525	3.568	3.652	3.747	3.789	3.838	3.943
5	3.196	4.279	3.931	2.711	6.156	3.605	2.547	3.747	3.838	3.912	4.008
6	7.231	5.297	5.303	2.409	1.823	1.855	1.841	1.869	3.895	1.931	1.931
7	5.24	5.302	5.323	5.372	1.801	1.809	1.831	4.85	1.864	1.857	1.942
8	5.267	5.291	5.364	5.418	1.795	1.839	1.838	1.862	1.937	1.881	1.923
9	5.263	5.263	5.344	5.418	1.788	1.79	1.871	1.906	1.868	1.911	1.949
10	5.263	5.316	5.354	5.42	1.793	1.801	1.858	0.787	1.876	1.907	1.94
11	5.221	5.323	5.393	5.401	1.828	1.816	1.87	1.856	1.887	1.904	1.924
12	5.26	5.306	5.315	5.4	1.826	1.799	1.84	1.888	1.887	1.908	1.929
13	3.269	5.32	7.313	5.391	1.783	1.826	1.803	1.864	1.869	1.895	1.922
14	5.249	5.304	5.356	5.397	1.829	1.81	1.845	1.849	1.883	1.907	1.915
15	5.262	7.133	4.336	5.393	1.824	1.786	1.878	1.886	1.881	1.896	1.926
16	5.235	5.315	5.349	6.388	1.801	1.845	1.872	1.854	1.896	1.902	1.933
17	5.268	3.128	5.343	5.385	2.775	1.053	1.836	1.899	2.313	1.896	0.947
18	5.207	5.295	4.369	5.403	1.82	1.789	1.849	0.897	1.903	1.905	1.912
Years	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011

TABLE 1. Rutting values(mm) for a 1.8 kilometre section over eleven years.

Length	Known PGs	F Sets	PGs Found	Seconds
29	3	4	3	0.75
30	3	6	3	1.2
35	4	7	4	1.5
40	4	6	4	1.8
78	5	9	5	2.5
90	6	11	6	3.1
120	15	19	15	4.1
160	7	28	7	5.3
200	13	29	13	7.2
215	9	17	9	3.8
260	11	18	11	4.6
310	15	11	15	7.6
365	12	21	12	8.3
400	15	27	15	6.4
415	16	23	16	12.6
470	10	18	10	5.3
509	18	36	18	11.5
545	13	29	13	5.6
604	21	31	21	9.25

TABLE 2. Performance of the meta–analytic technique on a selection of simulated sections of various lengths with predefined progression groups (PGs), showing the number of founder sets (F Sets) found with $\mathcal{R}$ as the Pearson correlation coefficient, the number of progression groups discovered, and the time taken in minutes to complete the analysis.

List of Figures

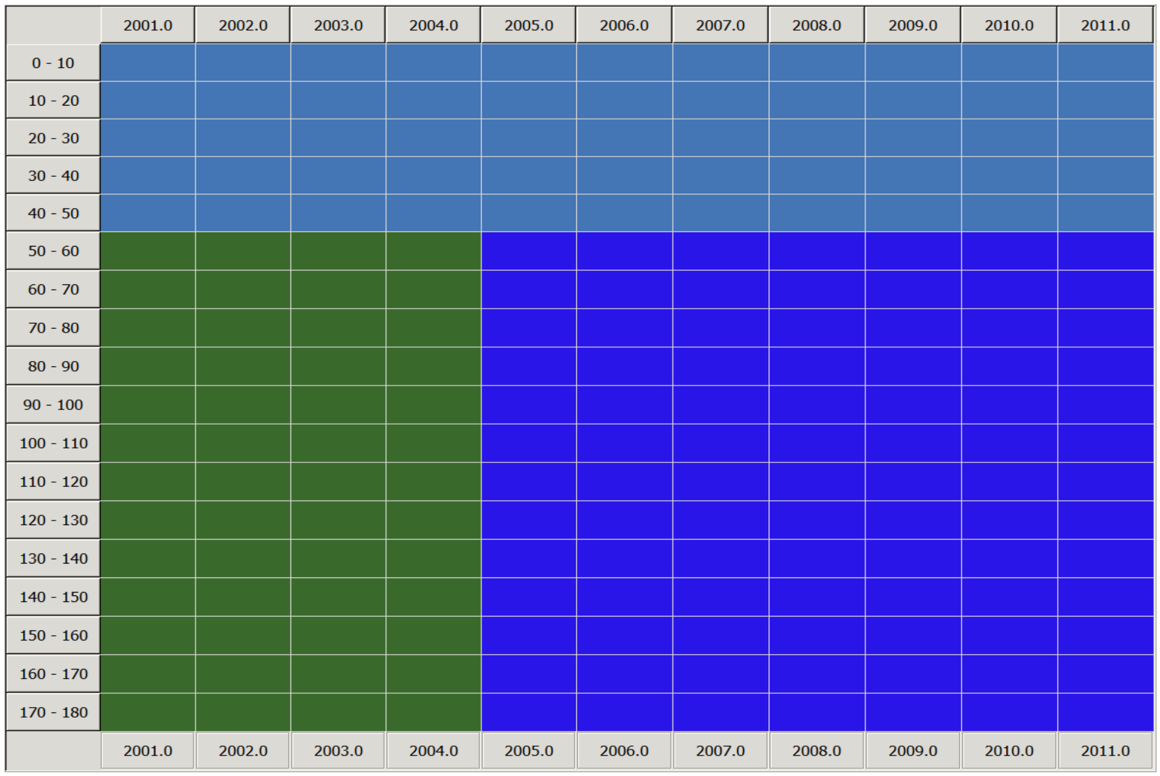

FIG. 1. Progression groups for the example section in Table. 1. There are two progression groups: (i) from 0 t0 50 meters and (ii) from 50 to 180. The position of maintenance interventions and progression groups are shown in coloured blocks at the left, whereby each block is a group of adjacent 10 meter chains that share the same progression rate.

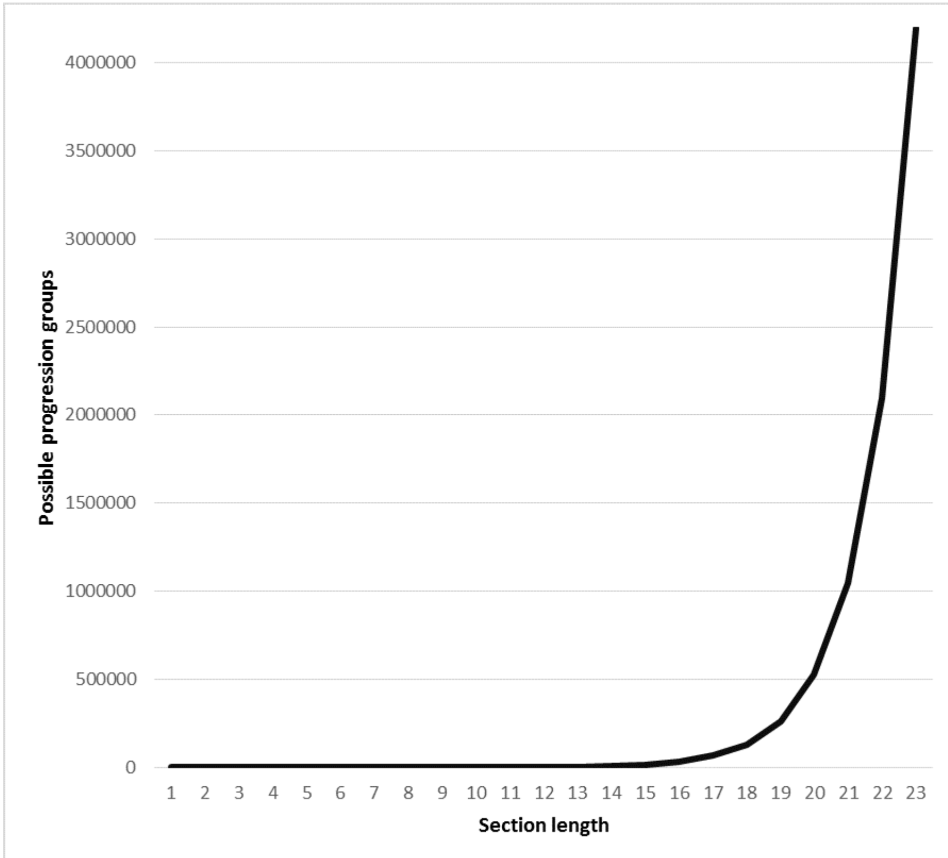

FIG. 2. Increase in the number of possible progression group models in relation to section lengths. Section lengths are on the horizontal axis while the number of progression group models that can be generated are on the vertical axis.

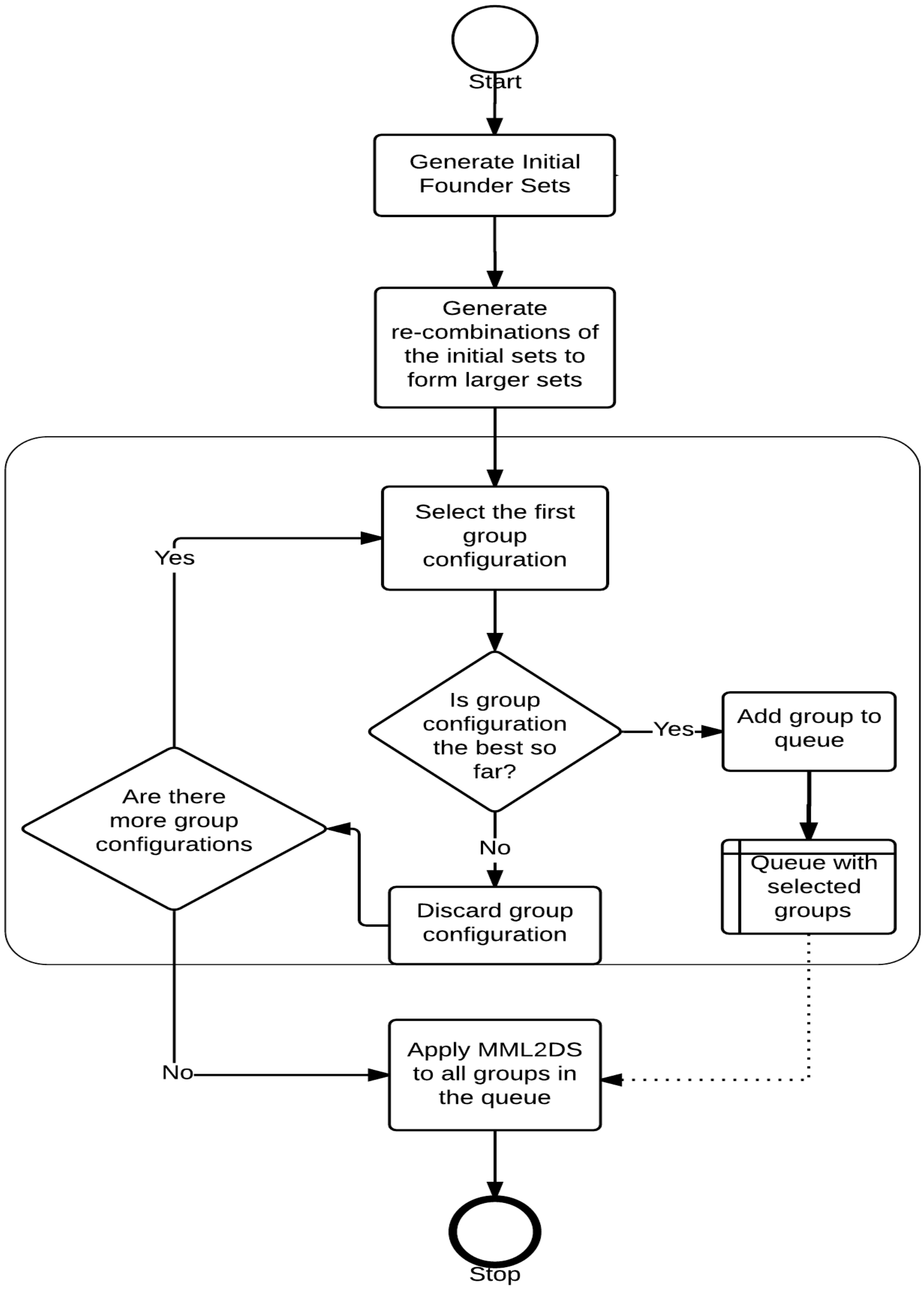

FIG. 3. Flowchart depicting the meta–analytic procedure applied to a section.

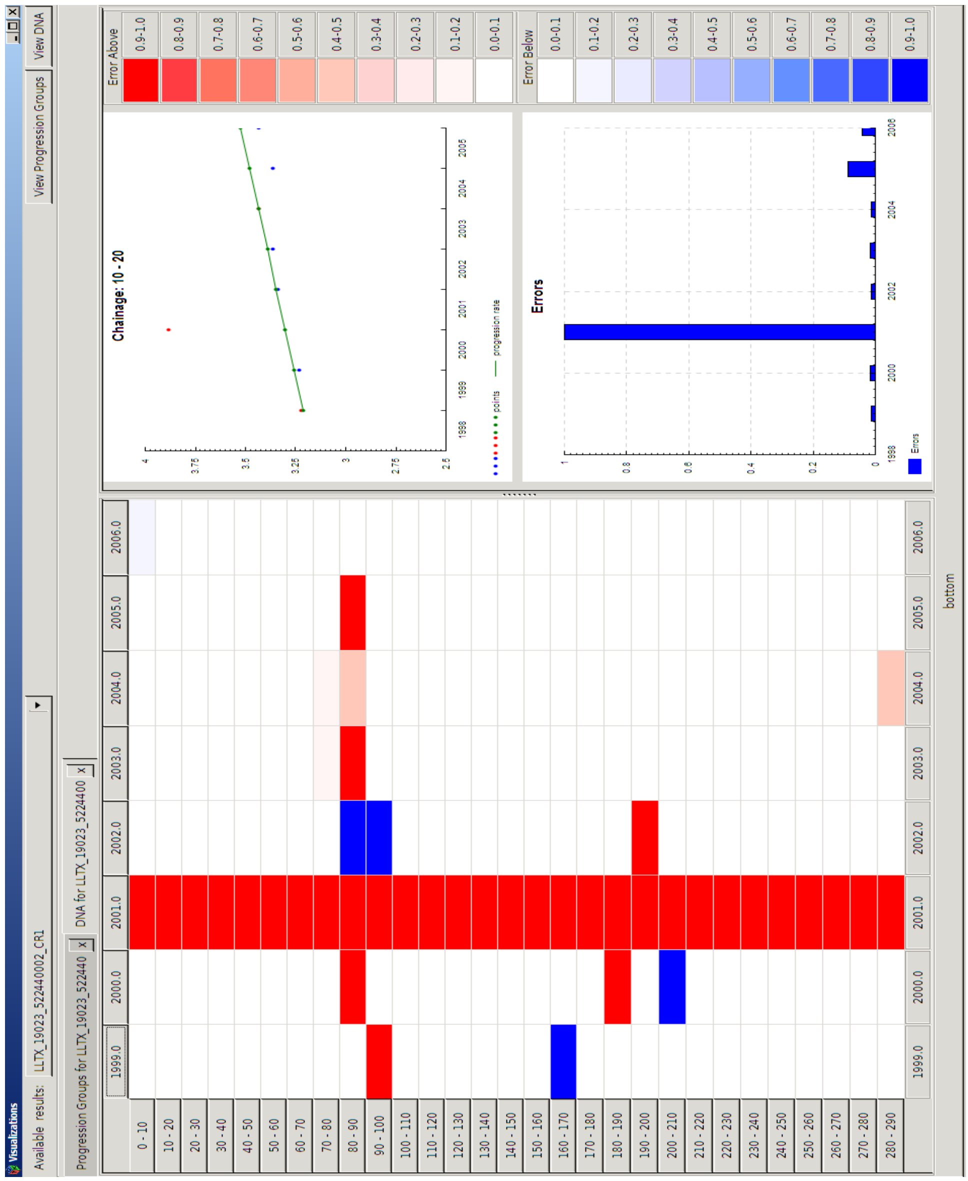

FIG. 4. Progression rate and error.

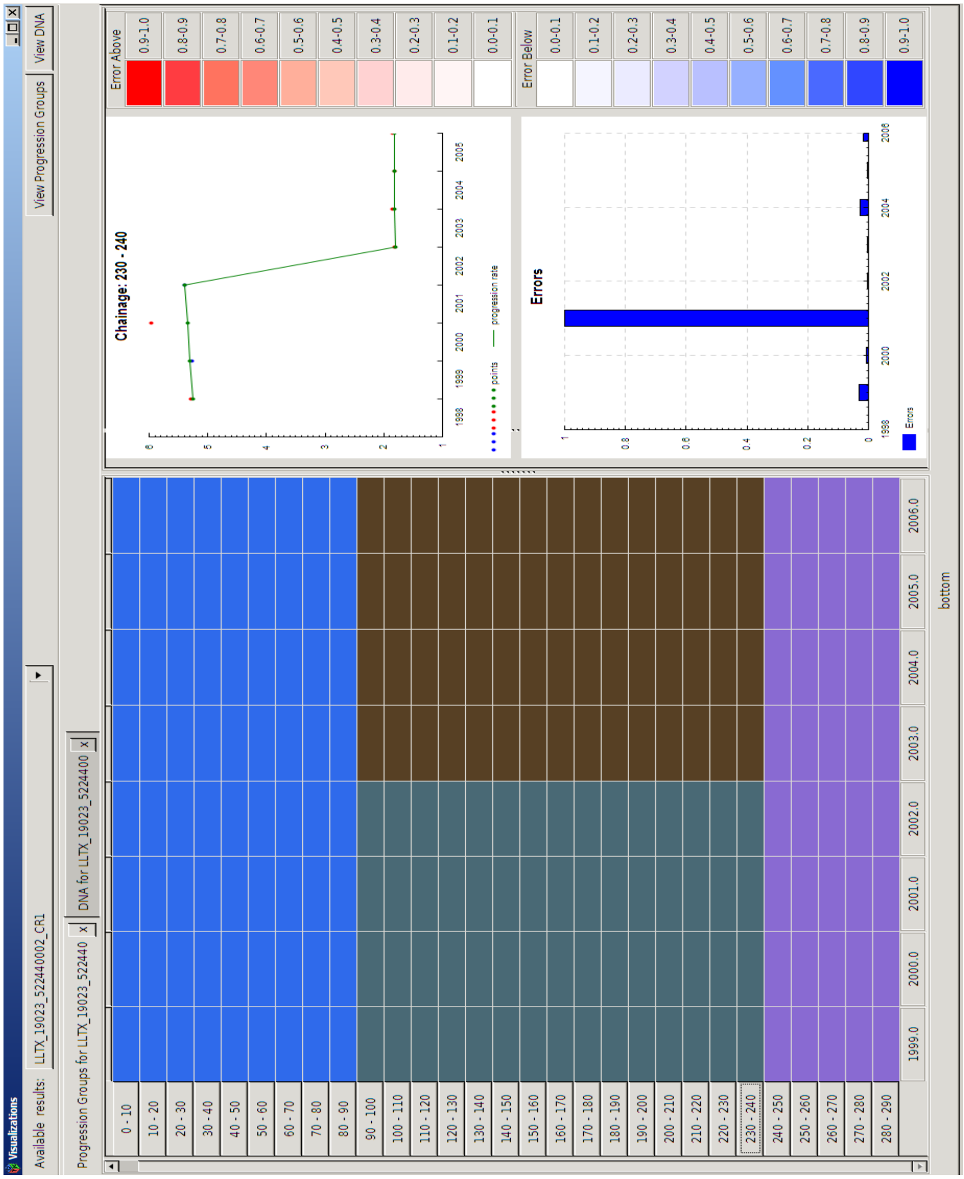

FIG. 5. Progression groups identified on a section with the fitted progression rates and maintenance intervention patterns. There are three progression groups: (i) from 0 **t0** 90 **meters, (ii) from** 90 **to** 240 **meters, and (iii) from** 240 **to** 290**. The position of maintenance interventions and progression groups are shown in coloured blocks at the left, whereby each block is a group of adjacent** 10 **meter chains which share the same progression rate. Chain** $230-240$ **has been selected, showing a clear maintenance intervention occurring between years** 2004 **and** 2005 **and this intervention pattern exists across all chains from** $90-100$ **to** $230-240$**.**