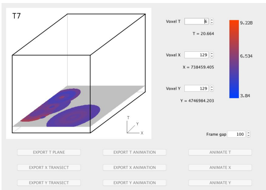
TIMESCAPE LOCAL
SPACE-TIME INTERPOLATION TOOL: Projected Coordinates Java Standalone Application

Marco Ciolfi,[a] Francesca Chiocchini,[a] Michele Mattioni,[a] Marco Lauteri[a]



Many ecological variables vary both in space and time and the datasets are often distributed far form the ideal statistical sampling: the Timescape algorithm attempts to cope with environmental sciences datasets with a scattered spatial and temporal distribution. TimescapeLocal, in particular, is suited for the use of projected coordinates datasets, including the widespread Universal Transverse Mercator and Lambert Conformal Conic spatial reference systems (local to regional scales). The spatial distances can be evaluated according to the standard Euclidean, diamond and square metrics, then a spacetime distance, which includes a causal constraint, is computed. The interpolation then follows, with a few ordinary spatial interpolation algorithm, included in the software package. A typical run lasts a few hours, depending on the complexity of the algorithm, the metrics chosen and the number of output cells.
Keywords: Software, Java, projected coordinates datasets.

0 Introduction

The notion of spacetime in physics dates back to the beginning of the 20^{th} century with the Minkowski formulation of the Special Relativity Theory. Since then, the spacetime approach has been proven efficient in field theory and statistical physics as the ideal aren for the description of the most diverse phenomena. Spacetime is the stage of changing patterns in many other disciplines and we believe that it could be useful in forest ecology modeling as well, despite some mathematical subtleties.[1]

Timescape borrows from the spacetime of Physics the notion of causality.[2] In relativistic Physics the notion of *light cone*, strictly linked to causality, emerges from prime principles; this is not the case in the spacetime described above, with the ordinary Euclidean distance. In our case a causal structure has to be "enforced by hand", imposing a constraint of maximum possible influence. We define the *causal ratio* r_{AB} as the adimensional ratio between the ground distance and the time distance of the points A and B, i.e.

$$r_{AB} = \frac{\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2}}{c\,|t_A - t_B|} \tag{1}$$

this quantity measures how far (in space) A and B are with respect their separation in time. The factor c is needed to keep the quantities comparable. The figure 0.1 below shows the space dimension(s) horizontally and the forward-only time dimension vertically.

Enforcing a causal constraint means defining a certain cone of influence on such abstract space. A *forward causal cone* contains all the possible outcomes of a source point x (red dots), while the yellow dots indicate unreachable events, i.e. they cannot have been influenced by x. In the same way, green dots belong to the *backwards causal cone* so they are possible causes for x, while the blue dots are not. The aperture of the cone can be adjusted according to the very nature of the modelled phenomenon, so for any point x of the model at a time, say, t_0, we have a moving forward surface S_x^+ which scans the future of x at any time $t^+ > t_0$ and a backwards surface S_x^- which scans the past of x at any time $t^- < t_0$.

Inflating the two-dimensional construction to a full three-dimensional representation, as in the figure above, we can see that every point x of the model possesses a causal structure. In particular, following the same colour coding, the green dots are the set of events (a subset of our observations) which could have caused x, so we must estimate the our value at x using only its green dots, i.e. the elements falling into x's backwards causal cone.

How wide should the cone be? it is controlled by the Timescape parameter k, which is the maximum acceptable value r_{AB} for two points A and B to be causally connected (eq. 1). Operatively we have just to check $r_{AB} \leq k$; which point is the cause and which

[a] *Istituto di Biologia Agroambientale e Forestale (IABF), Via Guglielmo Marconi N. 2, Porano (TR), Italia.*

1 The Minkowskian spacetime is not what is going to be used in Timescape. Nor Timescape has anything to do with Relativity, even if there are -deliberately- some common mathematical techniques involved.

2 Causality emerges naturally in a relativistic framework as a consequence of a maximum allowed velocity, the speed of light c, thus dividing the spacetime of a point in accessible (so called *timelike*) areas and unreachable (*spacelike*) ones, even for the light itself.

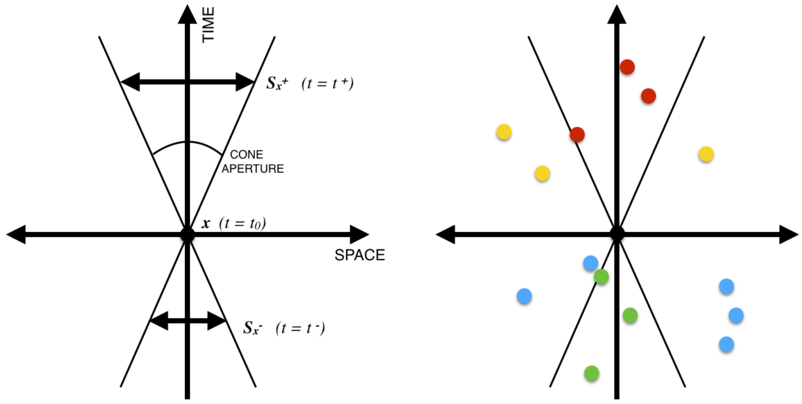


Fig. 0.1

one is the outcome depends on the values of time:

$$\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2} \leq k\,c\,|t_A - t_B| \tag{2}$$

Large values of k are related to loose constraints, i.e. a fast diffusion is allowed.[3] on the other hand, small values of k are related to very strict causal constraints, that is low diffusion rates. The value of k is assumed constant here for the sake of simplicity, but it can be a function of the position in the spacetime and also of ancillary variables (external conditions).

The particular flavour of a Timescape model is given by the combined values of c and k. c tells us how much we must correct our readings with respect to time differences, while k says how strict we are about causality.[4]

0.1 Model Evaluation

The Timescape model is built as follows: an empty model is inserted into the database then, one *sheet* (constant time section) at a time, all the voxels are evaluated. Each sheet's voxel has its own causal cone which contains only a subset of all the samples from which we are interpolating. If the cone is empty, no value can be attributed to the voxel. After the interpolation of the voxels' values, these are re-inserted into the database, and the calculation proceeds with the next sheet, till the last.

If the backwards causal cone of the voxel located at a certain point x contains at least one source point, the evaluation can take place as follows:

- Distance assessment: for all the source points s_k evaluate the corresponding distances d_k from x, thus obtaining the couples (s_k, d_k). Each source point s_k possesses a value v_k.

- Pruning: order the couples by increasing distance and retain only the first N of them. It is possible to skip this phase and use all the couples. Usually just the few nearest s_k give a significant contribution.

- Spatial statistics: use any established geostatistical algorithm to estimate the value at x. For example, using a simple IDW, where the weights $w_k = 1/d_k$, the value v at x is:

$$v = \frac{\sum_k w_k v_k}{\sum_k w_k}$$

The last step ensures that Timescape is at least as good as the spatial interpolator adopted, inheriting from it the proof of convergence (and the interpolator's defects too, of course). As of now, there are lots of options for the spatial statistics step, other than the plain IDW: **TimescapeLocal** offers a variety of methods, ranging form IDW to Kriging,[5] incorporating also harmonic (periodic) corrections.[6]

In fact, the Timescape Algorithm does not act on interpolation itself, but on the structure of the samples dataset with a suitable distance definition. It is like reshuffling the dataset for each interpolated voxel, so that each voxel is a little model on its own. This procedure is fair as long as the distances defined are true distances.

3 An infinite value of k is acceptable as well, meaning that the causal cones fill al the space.

4 Technically speaking, it is a Wick-rotated Euclidean version of Minkowskian spacetime but, unlike the latter, it requires two separate parameters to establish the causal relationships among points. This construction resembles the *light cones* of Special Relativity: as said, this is not by chance, but the realms of possible applications of the two constructions are completely different. Timescape is thought for ecological issues. The interested reader can consult any manual about Special Relativity. The technical term for a point of spacetime is *event*, but we will not follow this convention here due to possible confusions with the layman meaning of the word *event*.

5 The kriging evaluation is still in-progress code, it is marked as –TODO– in the appropriate software panel.

6 It is possible to plug one's own interpolators, but it requires the coding of a Java class per each new interpolator, extending a suitable abstract class. Reflection is used in order to incorporate automatically new user-defined classes; sometimes it causes some trouble, depending on the user's Virtual Machine settings.

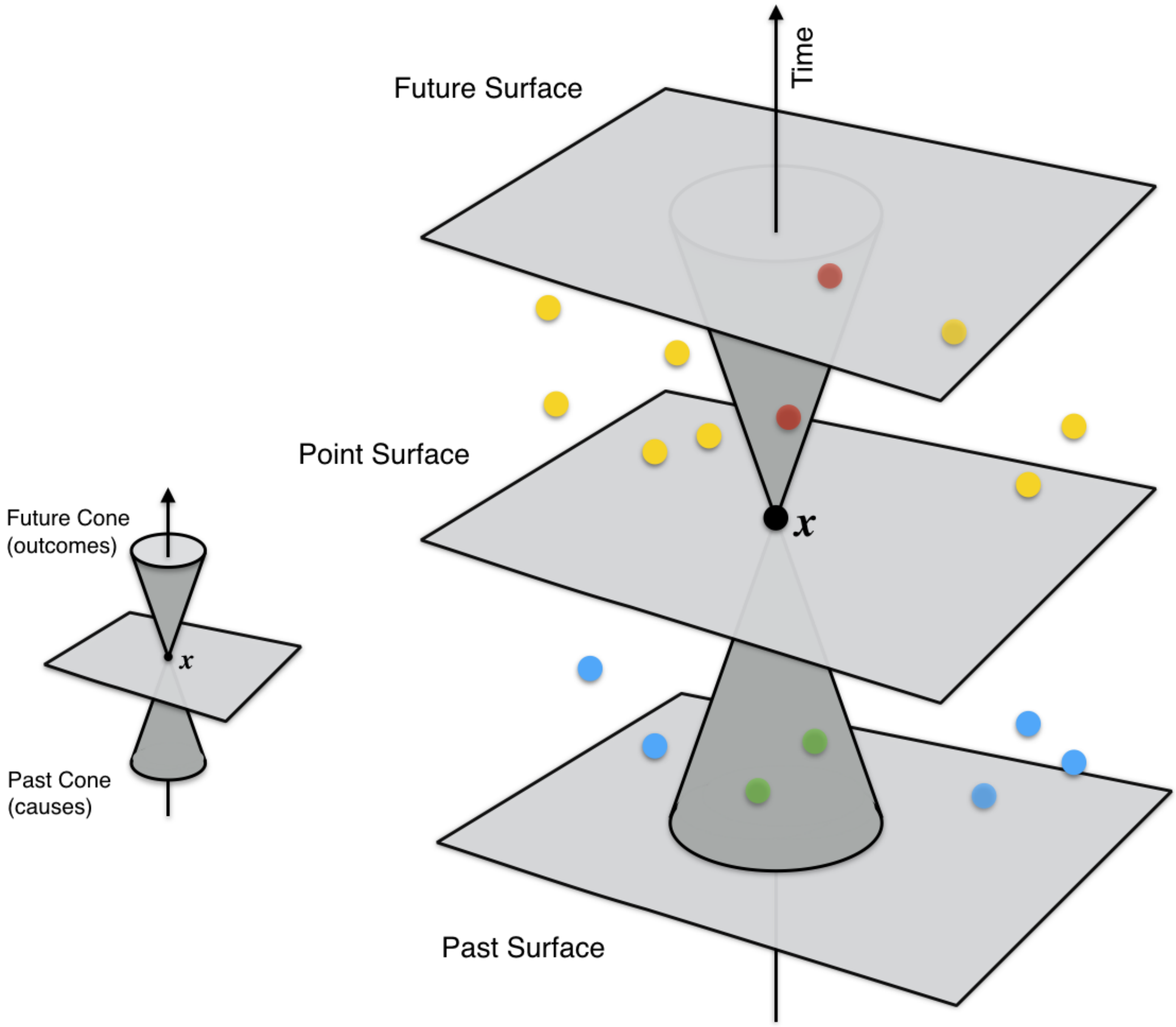


Fig. 0.2

0.1.1 Model Tuning

There are many options for Timescape tuning, some were already mentioned before, about distances and causality. The distinctive character of any Timescape model is given by c and k, since these model the space-time relationship according to the user's needs, but how to choose these parameters if there is no clue about their values? As a rule of thumb, as an initial guess c should be chosen in such a way that the model bulk is roughly a cube, i.e. $c\Delta T \sim \Delta X$ or $c = \Delta X/\Delta T$, where ΔX and ΔT are the space and time intervals of the model; if they are not comparable, probably space or time variability is the prevailing aspect by far, so one can use ordinary spatial statistics or time series analysis techniques. A first guess for k is harder to motivate, one can simply try $k = 1$ and move towards ∞ or 0 if there are too few or too many points falling inside the causal cones.

If one knows about a transport phenomenon which is occurring about the variable under scrutiny, c can be chosen as the appropriate velocity (in units of length per units of time.[7] A good guess for k should be 2 or 3; $k = 1$ can be chosen if one knows for certain that c is an insuperable threshold.[8]

The fine tuning of a model is achieved through a set of other parameters, which include:

- The *neighbourhood consistency* N: it is the maximum number of source points to be considered for the statistical interpolation.
- The *maximum distance* D, if we assume that the source points more than D apart cannot have any influence on the estimate.
- The *metric* employed: the neighbourhood of a point can considered a circle a square or a diamond.[9] It does not affect the general behaviour of a model but can change the values.

It is also possible to interpolate the values from ancillary variables instead of the actually measured values, or to corroborate the

7 There are no predefined units in **TimescapeLocal**, space can be given in metres, as a general rule, for this is tue unit of UTM and Lamber Conical projections.
8 This is the case in Special Relativity, where c is the speed of light and no other parameter is needed to define causality.
9 These are said to be *equivalent metrics*.

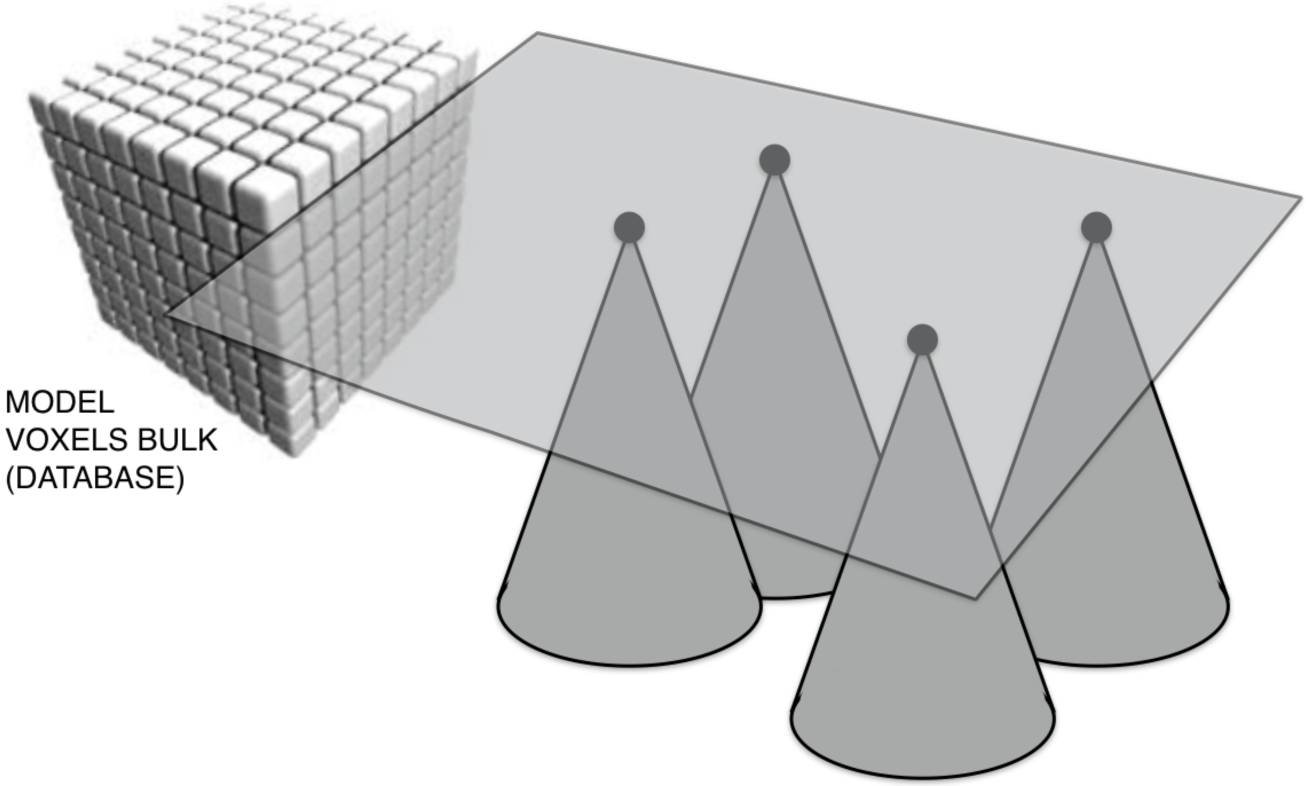


Fig. 0.3

estimates using both measured and ancillary values. Users need to define their own interpolation functions using a simple syntax.[10] A note of caution is in order here, however, since the constraints of a well-behaved distance have to be satisfied, or there is no guarantee that the spatial statistics methods employed can converge.

The number of voxels which constitute the Timescape model can be defined, ranging from a few thousands for little downscaled models to billions, depending on storage capacity.

0.2 Model Storage and Exploration

Admittedly, a real drawback if Timescape is the need of a lot of room for model storage. This calls for an external database. An external storage, however has its advantages too. The model can be queried in standard SQL[11] (Structured Query Language) for any need that users can have. To keep the bulk to a minimum the actual coordinates of the voxel are not stored into voxels table, which contains only their references, i.e a voxel is represented as a record

$$\text{voxel} \;=\; \left[k, i, j; \text{value}\right]$$

where (i, j) label the space site (the "horizontal" coordinates) and k labels the time sheet (the "vertical" coordinate). It is up to the user to get back to the real-world coordinates and time.

A voxel record is so composed of three integer numbers, the coordinates pointers, and a real number,[12] the value. The indices (i, j, k) range from zero for minimum x/longitude, minimum y/latitude and time, and their maxima N_x, N_y and N_t, corresponding to the maximum value of the coordinates. The total number of voxels is $N_xN_yN_t$, which corresponds roughly to a size of $20N_xN_yN_t$ bytes. Just to give an idea, a small $100 \times 100 \times 100$ model needs about 20 MB and a $1000 \times 1000 \times 1000$ one requires 20 GB of database space. This is the main reason for choosing an external database for models' storage.

The Timescape published software versions offer a variety of pre-packaged querying tools. These include statistical analysis tools and allow the export of different subsets of the model.

Timescape can be thought of as a "detour" from the standard users' flow, combining GIS and statistical analysis. To this end, it is possible to export the time sheets as GRID files, a common GIS raster standard (they are human-readable ASCII files), and cores dug at at given site, which are in fact time series of modelled values.

10 As of now, functions must be implemented in Javascript, which is a fast and easy to use also for non-expert users. Most algebraic functions can be implemented as-is in textbook notation.

11 The standard distribution of **TimescapeLocal** is based on a MySQL database. The database connection, however, is mediated by a Hibernate framework, so users can adopt any Hibernate-compatible database.

12 Technically, the value is a so-called *double* number.

It is also possible to export data in form of *.csv* files (comma-separated formatted ascii files). This is the standard input of any serious statistical package; *.csv* files can be imported in most spreadsheets, too, but the quantity of records discourages such approach for the bigger models.

1 The Timescape Maths

This rather technical section describes in detail all the maths underlying the Timescape Algorithm and its possible implementations. Skip to the next section for hands-on software installation and usage instructions.

1.1 Space and Time Distances

The question of what should be considered a measure of distance in a spacetime is not a trivial one. The Minkowskian spacetime of Physics is centred on the invariance of the speed of light and it is not suitable for the kind of problems encountered other sciences' datasets (i.e. ecology); we will introduce a set Euclidean or pseudo-Euclidean measures which suite our needs. There is nothing "relativistic" in Timescape, although its construction resembles (and in fact it is borrowed from) the Minkowskian double cones. In fact, we will not define the distance, but rather a set of suitable distances.

Following the current Minkowskian terminology, we call an **event** $x = (t, \mathbf{x})$ any point of spacetime X, where t is the time coordinate and $\mathbf{x}$ are the spatial coordinates. X has not to have a definite topology. It can be simply $\mathbb{R}_0^+ \times \mathbb{R}^2$ (flat space) or something topologically equivalent to $\mathbb{R}_0^+ \times \mathbb{S}^2$ (a sphere, an ellipsoid or something more general, too).[13]

Starting from the time, the distance d_t is very simple to define. Given two events $x = (t_x, \mathbf{x})$ and $y = (t_y, \mathbf{y})$, the distance inherited from $\mathbb{R}$ could be defined as $|t_x - t_y|$ which can be made homogeneous with the space components by multiplying it for a parameter c having the dimensions of a speed: $c\,|t_x - t_y|$. Or, more generally, we allow c to be a function of the events:

$$d_t(x,y) \;=\; c(x,y)\,|t_x - t_y| \tag{3}$$

The function c in (3) should be constrained in order to obey all the constraints for a distance function (4). Every positive constant will do, as some monotonously increasing functions can do, but not all of them (the triangle inequality could fail to be true). Constraints-violating functions can be used, but this "breaks the rules" somehow. A remarkable exception is the use of harmonic functions in c: these are obviously not well-behaved distances but can be of great help in modelling periodic phenomena.

The spatial part of the distance $d_s(x,y)$ is more complicated to define. It depends on the topology of the space, so the use of a Riemannian metric is due, in principle. The point is that evaluating the geodesic distance between two events is time-consuming and in most of the cases not worth the effort. As a general rule, a distance should satisfy the following constraints:

$$\begin{aligned}
& d(x,y) \geq 0 \;\; \forall\, x,y \in X \quad \text{non-negativity} \\
& d(x,y) = d(y,x) \;\; \forall\, x,y \in X \quad \text{symmetry} \\
& d(x,y) = 0 \;\; \text{iff} \;\; x = y \quad \text{coincidence} \\
& d(x,z) \leq d(x,y) + d(y,z) \;\; \forall x,y,z \in X \quad \text{subadditivity}
\end{aligned} \tag{4}$$

The formal solution consists in measuring the geodesic line from x to y (or vice versa, the result does not change for symmetry). This is achieved solving the equation for the path $x(\tau)$ parametrised by τ[14] (sums over repeated indices understood):

$$\ddot{x}^{\lambda} + \Gamma^{\lambda}_{\mu\nu}\,\dot{x}^{\mu}\,\dot{x}^{\nu} \;=\; 0 \tag{5}$$

where $\dot{x}^{\lambda}$ and $\ddot{x}^{\lambda}$ indicate the first and second derivatives of the λ component of x relative to τ and $\Gamma^{\lambda}_{\mu\nu}$ is the Christoffel symbol related to the metric. Numerically solving (5), though feasible in principle, is not compatible with acceptable running times, at least not with a standard desktop hardware.[15] Compromises are in order, here. The shape of the Earth can be approximated with an oblate ellipsoid, for which exact solutions exist, but they involve te evaluation of elliptic integrals; these can be evaluated numerically, but it is still too much for an ordinary computer in a reasonable time. On the other hand, the use of projected coordinate in **TimescapeLocal** is straightforward in that we can use a simple Euclidean sum.

If we have a flat N-dimensional space the distance can be the simple Euclidean one:

$$d(x,y) \;=\; \sqrt{\sum_{k=1}^{N} \left(s_x^k - s_y^k\right)^2 + c^2\left(t_x - t_y\right)^2} \tag{6}$$

where s^k is the k-th space component of a point. Aside from a factor c, this is the ordinary Euclidean N-dimensional distance, not

13 $\mathbb{R}$ is the set of real numbers, $\mathbb{R}_0^+$ is the set of positive real numbers, including the zero value and $\mathbb{S}^2$ is the two-dimensional sphere. With *sphere* here we mean the layman's surface of the sphere only.

14 A sort of proper time, in the relativistic jargon.

15 The calculation has to be repeated a huge number of times: the number of space cells of the model times the number of source points: something easily of the order of magnitude of billions.

to be confused with the Minkowskian relativistic distance, which is different for having a sign switched between the spatial and temporal parts.

It is worth mentioning that more distances can be used other than the Euclidean one. In fact we can use the equivalent diamond metric

$$d(x,y) \;=\; \max\left\{ \max_k \left\{ \left|s_x^k - s_y^k\right| \right\}, c\left|t_x - t_y\right| \right\}$$

or the square (the so-called Manhattan) metric

$$d(x,y) \;=\; \sum_{k=1}^{N} \left|s_x^k - s_y^k\right| + c\left|t_x - t_y\right|$$

or a fancier admixture of these, combining e.g. a Euclidean sum of d_t with the diamond/square version of d_s. All these metrics are equivalent to the Euclidean one.[16]

1.2 Causal Structure

The key feature of the Timescape Algorithm is *causality*. The spacetime structure described in the previous section is, aside from a factor c, just ordinary three-dimensional space. The time still needs to be singled out as the "direction" followed by the patterns of change. We have to plug a causal structure by hand into the spacetime in order to drive the change towards a *forward only* direction. The Minkowskian metric[17]

$$d_M^2(x,y) = c^2\left(t_x - t_y\right)^2 - \|\mathbf{x} - \mathbf{y}\|^2$$

has a natural interpretation in terms of causality: a positive $d_M^2(x,y)$ means that y falls inside the *causal double cone* of x, either in its past ($t_y < t_x$) or in its future ($t_y > t_x$), while a negative $d_M^2(x,y)$ means that there is no causal connection between y and x.

Our Euclidean spacetime is topologically different form a Minkowskian one, so the causal double cone structure cannot emerge in a natural way. We must impose the causal structure somewhat artificially. Let define a *cone aperture* k which, in principle, can be a positive function of the events $k(x,y)$ corresponding to the maximum acceptable ratio between the space and time components of the distance:

- y can be a *cause* of x if $d_s(x,y) \leq k\, d_t(x,y)$ and $t_y < t_x$
- y can be an *outcome* of x if $d_s(x,y) \leq k\, d_t(x,y)$ and $t_y > t_x$
- y has no causal connection with x if $d_s(x,y) > k\, d_t(x,y)$

The aperture of the cone, or of the cone-like structure if $k(x,y)$ is not a constant, defines the strictness of the causality constraints: the broader the cone (a big k) the looser the constraints; the narrower the cone (a small k) the stricter the constraints. This structure, unlike the rigid Minkowskian one, allows for the definition of a flexible concept of causality.

An infinite value of k corresponds to a double cone spanning all the spacetime, so that any event can be in principle connected with the all the others. On the other hand, $k = 0$ means that there is no spread of causality in space, a complete static solution where the influence is limited is limited to the *time line* passing through x. Allowing $k = k(x,y)$ means that the causal constraints can vary from place to place and over time.

Now we must find a way to translate mathematically the constraints mentioned above in an easily computable way. The key is the Heaviside theta or step function:[18]

$$\theta(t) = \begin{cases} 1 & \text{if } t \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

this function acts as an on/off switch so that

- y can be a *cause* of x if $\theta\Big(k\, d_t(x,y) - d_s(x,y)\Big)\, \theta\left(t_x - t_y\right) = 1$
- y can be an *outcome* of x if $\theta\Big(k\, d_t(x,y) - d_s(x,y)\Big)\, \theta\left(t_y - t_x\right) = 1$
- y has no causal connection with x if $\theta\Big(k\, d_t(x,y) - d_s(x,y)\Big) = 0$

A straightforward interpretation is that $k\, d_t(x,y)$ represents the maximum possible *area of influence* $A_x(t)$ of the event x at a given time t, while $d_s(x,y)$ is the actual spatial separation between such events. Whenever y falls into this $A_x(t)$, the contribution of x has to be taken into account, while when an event y' falls out of $A_x(t)$ (so out of the forward causal cone of x), the contribution of x is null.

---

16 Two metrics are sai equivalent when each ball in one metric contains at least a ball of the other, and vice-versa. A ball of radius r about an event $x = (t, \mathbf{x})$ being defined as the set of all events closer than r to x (an open ball), or closer or exactly r apart from x (a closed ball).

17 That, strictly speaking, is not a metric since it is not non-negative.

18 There is not consensus in the literature about the value of $\theta(0)$. Some authors put $\theta(0) = 1/2$, especially in the field of signal processing, for essentially practical reasons; here and in the following we adopt the definition of θ as the characteristic function of the $[0, +\infty)$ set, which cannot take fractional values.

A timescape model is just a lattice subset of the space of all the allowed events of the spacetime under scrutiny, equipped with the metric of choice. To every element (voxel[19]) corresponds a value of the investigated quantity.[20]

1.3 Building the Model

A Timescape model is a collection of voxels equipped with a value. The extent of the model is the product of a time interval $[T_m, T_M]$ times the spatial extent of the surface involved, say $[X_m, X_M] \times [Y_m, Y_M]$.

Each *time sheet* of voxels corresponds to a spatial replica at a given time. If M is composed by $N_T \times N_X \times N_Y$ elements, the kth time sheet has time

$$t_k = T_m + \frac{T_M - T_m}{N_T}\left(k + \frac{1}{2}\right), \quad k = 0 \dots N_T - 1 \tag{7}$$

with the $\frac{1}{2}$ bias correction factor to make the sheets match the centres of the corresponding plane of voxels. Following the same lines of surgery, every sheet is subdivided in pixels of centres (x_i, y_j) of coordinates

$$\begin{aligned} x_i &= X_m + \frac{X_M - X_m}{N_X}\left(i + \frac{1}{2}\right), \quad i = 0 \dots N_X - 1 \\ y_j &= Y_m + \frac{Y_M - Y_m}{N_Y}\left(j + \frac{1}{2}\right), \quad j = 0 \dots N_Y - 1 \end{aligned} \tag{8}$$

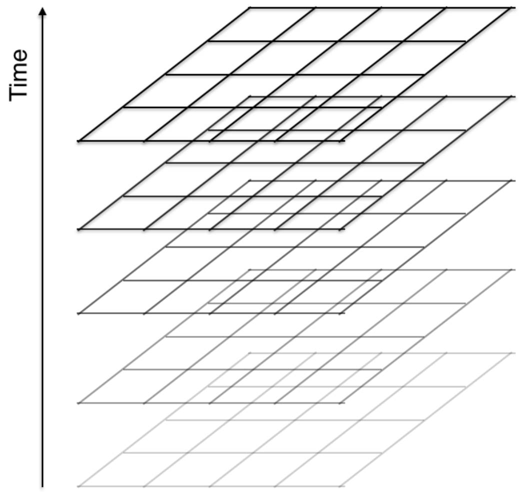


Fig. 1.1

A discrete Timescape model M consists of a finite set of voxel-value pairs m_{kij}

$$M = \{m_{kij}\} = \left\{(x_{kij}, v_{kij}) \mid x_{kij} \in \mathbf{ST}, v_{kij} \in \{null\} \cup \mathbb{R}\right\}$$

where x_{kij} is an event of the spacetime $\mathbf{ST}$ and v_{kij} is its value.

The *samples collection* is the set $S = \left\{(x_n, v_n, \mathbf{a}_n) \mid x_n \in \mathbf{ST}, v_n \in \mathbb{R}\right\}$ of samples at event x_n, with a value v_n and possibly an array $\mathbf{a}_n$ of associated ancillary variables.[21]

The *spacetime support* $\mathbf{S}$ of S is the finite set $\{x_n\}$ of the sample points events. In the following, with a slight abuse of notation, we will use S for both the samples collection and its spacetime support.

For any x_{kij} (x for short) we can interpolate a value, provided that there are candidate sample events: defining the weight function[22] $w(d; x, x_n, v, \mathbf{a}_n)$ and the estimate function $f(v; d, x, x_n, \mathbf{a}_n)$ we have for the value of x

$$v(x) = \left[\sum_{\text{causes}} w(v; d, x, x_n, \mathbf{a}_n)\right]^{-1} \sum_{\text{causes}} w(v; d, x, x_n, \mathbf{a}_n) f(v; d, x, x_n, \mathbf{a}_n) \tag{9}$$

where $d = d(x_n, y_{kij})$, of course. The normalised summation in (9) resembles a trivial inverse distance weighted estimation (IDW); this would be true if we choose the trivial $w = 1/d$ as the weight estimator, but this limitation can be easily overcome, by defining suitable weight functions. The summations in (9) are extended to the backwards causal cone of y_{kij}.

In the simplest case of the standard IDW interpolation, when the weight is given by $1/d^\alpha$, where α is a positive constant, (9) becomes

$$v(x) = \frac{\sum_{\text{causes}} \theta(kd_t - d_s)\left[\sqrt{d_t^2 + d_s^2}\right]^{-\alpha} f(v_n; \sqrt{d_t^2 + d_s^2}, x, x_n, \mathbf{a}_n)}{\sum_{\text{causes}} \theta(kd_t - d_s)\left[\sqrt{d_t^2 + d_s^2}\right]^{-\alpha}}$$

that represents an (almost) trivial IDW interpolation with a causal constraint added through the θ functions. Releasing the θ one obtains a simple three-dimensional spatial interpolation, missing entirely, of course, the causal structure. More general expressions all follow the general pattern

$$v = \frac{\sum \theta \times d^{-\alpha} \times f}{\sum \theta \times d^{-\alpha}}$$

19 A voxel is a discrete three-dimensional *volume element* just like a pixel is a two-dimensional *picture element*.

20 The value can be undefined (*null*) if the event represented by the voxel does not fall into any set of possible outcomes of the source events.

21 The v_n must be defined but the ancillary variables can be undefined.

22 The w depends basically on the spacetime distance of the events, and it can also be a function of the events positions (thus accounting for inhomogeneities and anisotropy) and the samples' value and ancillary values too. The f depends basically on the value of the sample (trivially the value itself) but can depend also on the other quantities.

1.4 Evaluation Algorithm

In order to build a model M we start from a the finite samples collection S which represent the actual observations (i.e. the physical measurements) of the investigated phenomenon and proceed as follows:

The first step consists in the definition of set of coordinates $\{(t_k, x_i, y_j)\}$ representing the events located at the centres of the discrete events which constitute M, given a *time sheet size* w_s and a *space cell size* w_c. These coordinates are grouped in the three arrays $\underline{t}$, $\underline{x}$ and $\underline{y}$, spaced apart a length w_c and a time w_s:[23] Operatively, all voxels are inserted empty in the database before their actual evaluation. This is done in order to minimise the database file growing / shrinking during the run. Voxels records will just be updated with the appropriate values after evaluation: this does not change the record size.

Algorithm 1 Preliminary steps

```
procedure COORDINATES(w_c, w_s, T_m, T_M, X_m, X_M, Y_m, Y_M)
    δt ← w_s
    t_0 ← T_m + δt/2
    while t_n ≤ T_M do                                  ▷ voxel sheets times t
        t_{n+1} ← t_n + δt
    δx ← w_c
    x_0 ← X_m + δx/2
    while x_n ≤ X_M do                                  ▷ voxel space coordinates x
        x_{n+1} ← x_n + δx
    δy ← w_c
    y_0 ← Y_m + δy/2
    while y_n ≤ Y_M do                                  ▷ voxel space coordinates y
        y_{n+1} ← y_n + δy
return {t, x, y}
```

Then the actual model is evaluated, one voxel at a time. As algorithm control "tuning knobs" we add three further parameters:

- A space distance threshold D_s so to discard from the summations all the events too spaced apart; this is like switching off any interaction for events farther than D_s.
- A maximum number of allowed near primes N_{max}: for any y_{kij} of M we use at most N_{max} elements of S for the calculations.
- A *picking probability* $\mathscr{P}$ which introduces some randomisation spicing to an otherwise deterministic procedure. Any x_n is included in the summation only with probability[24] $\mathscr{P}$ to allow the users check the stability of their datasets: The algorithm can be run many times and if the dataset is said to be *stable to* $(1-\mathscr{P})$ with precision σ if the variances of all the estimated elements of M are less than σ.

The most involved part of the evaluation algorithm is the selection of the causally connected sample points to be used in the summation loop. The next algorithm is very general and includes, as a matter of principle, the evaluation of the length of the geodesic line Γ connecting x_n and x. Though formally correct, this is not what it is actually performed during the calculations, which are specialised according to the spatial geometry.

A null value of d is associated with those events that are outside the backwards causal cone. Null-d elements are sorted down to the tail of K_x^- (the set of all possible causes of x).[25] The actual calculation of d_s depends on the geometry of the space.

This procedure has been designed in order to minimise the changes in terms of RAM occupation size of the objects used in the calculations. Operatively, a unique instance of all the source events is created once before the evaluation phase starts, then, for any element (the three outer loops) the causing events are put in reverse-distance order and optionally trimmed to a maximum number. All the events are recycled for the next voxel.

As said, any voxel is evaluated independently from the others, so any model can be easily subdivided according to a union of submodels, provided some care is given to the limits of the submodels and that cell- and sheet sizes are kept unchanged.

1.5 Advanced Stuff

For strengthening the results of the interpolations **TimescapeLocal** can be inserted into advanced statistical analysis procedures, based on the statistical mechanics concept of *ensemble*. A few possible implementations include the following:

1.5.1 Jackknifing

Depending on the consistency of the samples set S, it could be possible to subset (*jackknife*) the samples seeking for a bias-corrected estimate of the model values. This is achieved considering S as an *ensemble* of subsets S_k, so that $S = \cup S_k$ and, in general,

23 w_c is the *cell* spacing and w_s is the time-sheet separation.

24 A random number in $[0,1]$ is generated at each step of the summation loop and it it checked against $\mathscr{P}$.

25 This could be called the *backwards causal cone* of x. This construction resembles the Minkowskian double cones in the spacetime of Special Relativity, but it is in fact a bit more complex, in that the width of the cones (apex) is controlled by two parameters: c and k, which in general are not constants, but functions of the events.

Algorithm 2 Timescape Model

```
procedure MODEL(t, x, y, 𝓔, N_max, 𝓟, D_s, c, k, w, v)
    M ← ∅                                          ▷ initialise an empty model M
    for t_k ∈ t do
        for x_i ∈ x do
            for y_j ∈ y do
                x_kij ← (t_k; x_i, x_j)                ▷ the running voxel's coordinates
                v_kij ← null                           ▷ all voxels are created empty
                𝒦 ← ∅                                  ▷ initialise the set of possible causes of x_kij
                for x_n ∈ S do
                    if RAND ≤ 𝓟 then                   ▷ random picking, always true if 𝓟 = 1
                        d_t ← c(x_n, x_kij) |t_{x_n} − t_{x_kij}|
                        d_s ← ∫_Γ √g dτ                 ▷ geodesic length, or whatever
                        if d_s ≤ min{k(x_n, x_kij) d_t, D_s} then
                            𝒦 ← 𝒦 ∪ {(x_n, √(d_t² + d_s²))}   ▷ one more causal event
                if 𝒦 ≠ ∅ then
                    𝒦 ← SORT(𝒦, d)                     ▷ order by increasing distance
                    𝒦 ← TRIM(𝒦, N_max)                 ▷ keep only the closest N_max events
                if 𝒦 ≠ ∅ then
                    V, W ← 0                           ▷ loop dummy variables
                    for (x_n, d_n) ∈ 𝒦 do
                        V ← V + w(x_n, x_kij) v(x_n, x_kij)
                        W ← W + w(x_n, x_kij)
                    v_kij ← V/W                        ▷ a non-empty voxel value
    M ← M ∪ {(x_kij, v_kij)}                           ▷ append the new voxel to the model
return M
```

$S_k \cap S_{k'} \neq \varnothing$. Depending on the consistency of S this procedure can be performed or not (all S_k must be statistically significant). The standard Jackknifing procedure consists in correcting a biased estimate[26] using a collection of subsets S_k, each of which neglects only the element x_n:

Algorithm 3 Model Jackknifing

```
procedure JACKKNIFE(S)
    M = MODEL(S)                                   ▷ Evaluate the global model
    for n ← 1...N do
        M_n ← MODEL(S\{x_n})                        ▷ Evaluate the nth model
    for x_kij ∈ M do
        θ̂_kij ← v_kij                               ▷ Global estimator
        θ̂*_kij ← 0                                  ▷ Biased estimator
        for n ← 1...N do
            θ̂*_kij ← θ̂*_kij + v^(n)_kij / N
        θ̂_kij ← N θ̂_kij + (1 − N) θ̂*_kij            ▷ Correct the estimator of the global model
    return M
```

Jackknifing adds another factor N to the complexity of the calculations.

1.5.2 Reverse Parameters Estimation

Another smart trick allowed by ensemble techniques allows a sort of reverse modelling. If one knows not how to assign a value to the c and k parameters it is possible to use the Timescape Algorithm in order to find an estimate of such parameters. Since c and k are related to the transport/diffusion capabilities of the system, i.e. to the patterns of change of the investigated phenomenon, what we gain is, in fact, an estimate of such velocity.

We can think $(c,k) \in \mathbb{R}^+ \times \mathbb{R}^+$ as a space of parameters, for each (c,k) pair[27] we have a distinct behaviour of the model, it is possible that some values of c and k match the actual patterns of change better than others. To find these values it is customary to select a *control group* subsetting S into two disjoint sets: a sample set S_0 and a control set S_c.

Then we interpolate a set of models $\{M_n\}$, each of which corresponds to a parameters pair (c_n, k_n). From the model M_n we then evaluate the residuals according to S_c and try to minimise them. This is done evaluating the squares of the differences between the elements of S_c and the model-estimated values at the same events.

26 This does not correct a biased set of observations, of course.

27 Let's limit ourselves to the case of constant parameters. The general idea can be applied also to functions, but the complexity grows accordingly.

One thing we can try is a recursive adjustment of the parameters down to an error ξ:

Algorithm 4 Recursive Parameters Estimate

```
procedure ESTIMATE(S_0, S_c, ξ)
    X_c ← {(x_n^c, v_n^c) ∈ S_0}                ▷ The set of control events and associated values
    N ← ♯(S_c)
    E ← +∞                                      ▷ A trivially high error value
    while E > ξ do
        ADJUST(c, k)                            ▷ Find new parameters
        M ← MODEL(S_0)
        E ← 0
        for x_n^c ∈ X_c do
            v_n ← v_kij | d(y_kij, x_n^c) = min ▷ Estimated value
            E ← E + (v_n^c − v_n)^2
        E ← √E/N
    return M
```

This is an acceptable procedure if there is any clue about an $\text{ADJUST}(c,k)$, procedure otherwise it is just a random wandering in the parameters space. This is where *ensemble averages* (below) come into play.

1.5.3 Ensemble Estimation

We create a finite set P of tentative pairs of parameters $(c_n, k_n) =: p_n$ each of which corresponds to a model M_n (given the S_0 and S_c sets). We then evaluate all the models and the associated errors E_n.

Supposing that P exhaustively represents all the possible cases.[28] This can be extended to different pairs of positive functions $\left(c(x,y,v,\mathbf{a}),k(x,y,v,\mathbf{a})\right)$ which is a natural extension of simple constant values, but it complicates the already complex calculations beyond the reach of ordinary desktop computers.

The estimate goes as follows:[29]

Algorithm 5 Ensemble Parameters Estimate

```
procedure ESTIMATE(S_0, S_c, P)
    c̃, k̃, w ← 0                                ▷ Define new parameters and a global weight
    X_c ← {(x_n^c, v_n^c) ∈ S_c}                ▷ The set of control events and associated values
    N ← ♯(S_c)
    for (c_n, k_n) ∈ P do
        M_n ← MODEL(S_0, c_n, k_n)
        e_n ← 0
        for x_k^c ∈ X_c do
            v_k ← v_kij | d(y_kij, x_k^c) = min ▷ Estimated value
            e_n ← e_n + (v_k^0 − v_k)^2
        w_n ← N/√e_n                            ▷ A weight
        w ← w + w_n
        c̃ ← c̃ + w_n c_n
        k̃ ← k̃ + w_n k_n
    c̃ ← c̃/w
    k̃ ← k̃/w
    M̃ ← MODEL(S_0, c̃, k̃)                       ▷ Ensemble Model
return M̃
```

Also this procedure adds a significant amount of complexity to the calculations ($N+1$ models need to be calculated).[30] Nonetheless, the added benefit of obtaining an estimate of the dynamic parameters could shed a light on an otherwise obscure phenomenon. It is possible, however, to evaluate N downscaled, tiny models and then the full-scaled $\tilde{M}$ with the same parameters values.

It is advisable not to try to build a "smart set" of tentative parameters pairs, however. If one has an idea about their value, it is better to start a recursive seek. Otherwise, one can employ lattice techniques (sets of equally-spaced tentative parameters) or Montecarlo methods, i.e. randomly generated parameters, a lot of them, of course!

28 Here is *Ergodicity* hidden beneath the *exhaustively represents*. See J. P. Sethna, Entropy, Order Parameters, and Complexity, Clarendon Press, Oxford 2006, ISBN 0-19-856676-2.

29 In this case, S_0 and S_c need not be disjoint; as a limiting case they can be both S.

30 As is always the case with statistical ensembles, N should be a really big number.

1.6 Further Reading

Those interested in the inner gears of the algorithm can find some fine details of spacetime structure and modelling in (randomly picked):

- For a complete treatment of geometrical techniques in physics: Theodore Frankel, The Geometry of Physics, Cambridge University Press 2012. ISBN 978-1-107-60260-1.
- For a thorough examination of Minkowskian spacetime: Gregory L. Naber, The Geometry of Minkowski Spacetime, Springer 2012, ISBN 978-1-4419-7837-0.
- For a gentler introduction to spacetime issues (and much more): Abhay Ashtekar, Vesselin Petkov (Eds.), The Handbook of Spacetime, Springer 2014, ISBN 78-3-642-41991-1.
- For spatial and temporal statistical techniques: Noel Cressie and Christopher K. Wikle, Statistics for Spatio-temporal Data, Wiley Series in Probability and Statitics, John Wiley & sons 2011. ISBN 978-0-471-69274-4.

2 The TimescapeLocal Software

TimescapeLocal is the local flavour of Timescape Algorithm.[31] It is the second branch of the project, which operates on projected coordinates. The software does not take care of the actual projections used, as long as they are compatible with the distance functions provided (Euclidean or equivalent). UTM (Universal Transverse Mercator) or Lamber conic projections work well, as many other less used coordinate systems. The output coordinates values follow exactly the input ones, for the sake of compatibility with other georeferenced data that the user will have to work with in his/her project.

2.1 Installation

2.1.1 Distribution Package

The software distribution package consists in the following files:

- `Timescapelocal.pdf` this manual.
- `Timescapelocal_src.zip` the zipped source code.
- `Timescapelocal.jar` the program executable file (must be configured).
- `TimescapeConfigurator.jar` a little configuration utility program.
- the mysql directory containing:
  - `db.sql` the mysql database creation script.
  - `db_example.sql` same as above, with a preloaded sample dataset, including a (tiny) evaluated model and a blank copy of it.
  - `data.csv` an example dataset.

2.1.2 Creating the Database

The first step is the creation of the database. **TimescapeLocal** has been developed and comes equipped with a mysql database. Since the data layer of the application is mediated by hibernate,[32] it is possible to use any other compatible RDBMS other than mysql.[33]

The Database is structured so that only one table (`voxel`) is storage-critical; its records are kept as small as possible, while all the common detail are stored into the *Model description* tables. The *Source Points* block contains the source Dataset.

The rule is: one source dataset = one database. So to be able to use different datasets users have to create different databases. A creation script is available (`db.sql`) is available to set up a blank database. Use the example version (`db_example.sql`) to explore the software a while before putting in your serious dataset.

Data must be prepared for input as a comma separated value. When the application starts, it asks for the dataset, if the database is empty. The data have to be prepared according to the `ID, T, X, Y, VAL` format, where:

- `ID` is a unique sample id (alpha-numerical, but keep it simple, no strange characters!).
- `T` is the value of the "vertical" time coordinate.
- `X` and `Y` are the values of the "horizontal" space coordinates.

31 The global coordinates version, **TimescapeGlobal**, also released under GP:3.0 license, is available for download at `https://sourceforge.net/projects/timescapeglobal/`.

32 See `http://hibernate.org` for details.

33 See `https://www.oracle.com/mysql`.

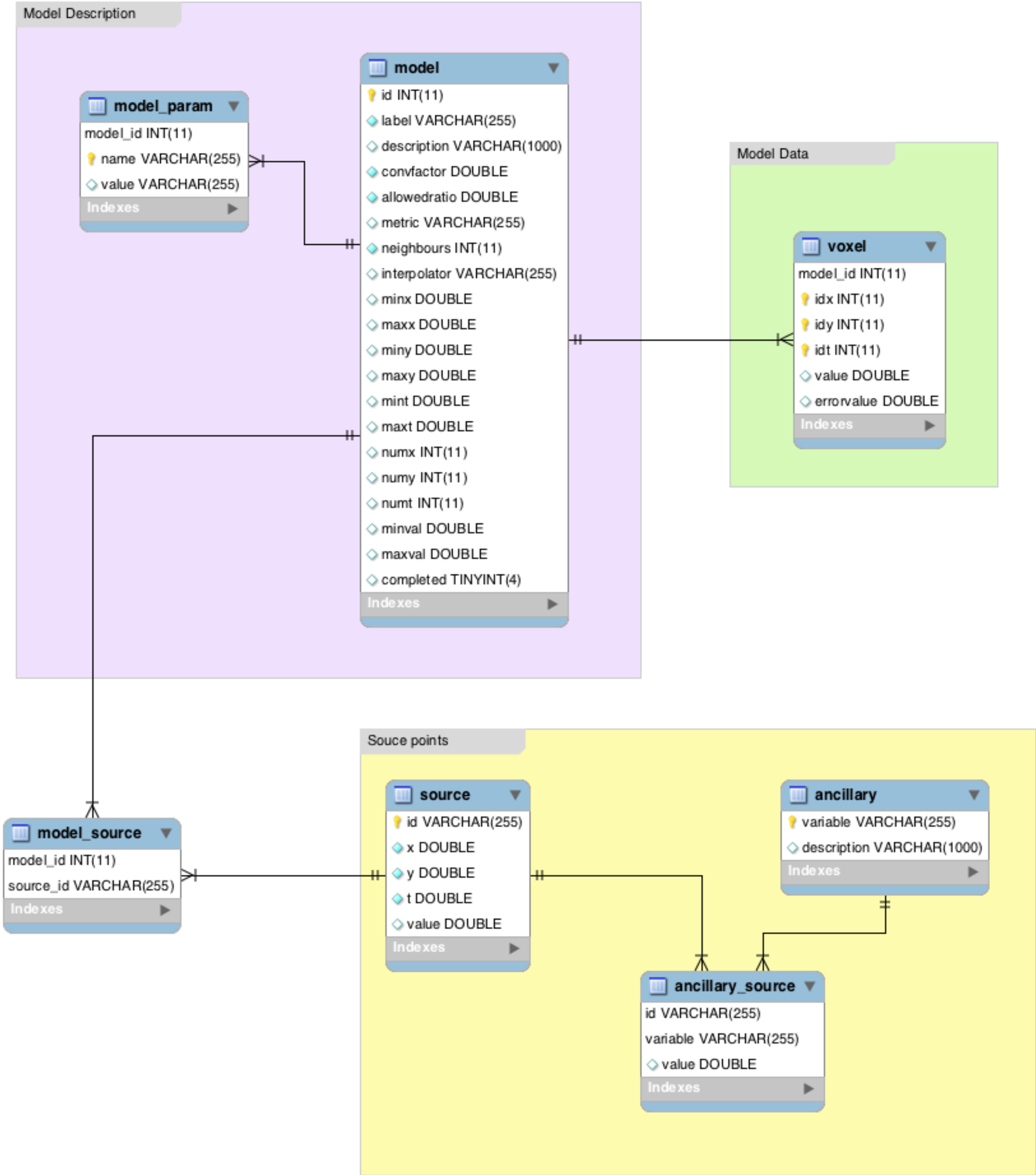


Fig. 2.1 Database ER model

- `VAL` is the value of the sample.

It is possible to append ancillary values, too. To do so, just append their names to the first line (say `ID, T, X, Y, VALUE, ANC1, ANC2, MYVAR`) also appending the corresponding values at the end of the data lines. Ancillary values can be empty. The ancillary variables names cannot be like the other fields names, of course.

2.1.3 Configuring the Program Executable jar

A few parameters need to be configured to use **TimescapeLocal**.[34]

It is possible, in principle, to perform some surgery on the jar itself, but it is safer to use the companion utility **TimescapeConfigurator**, that finds the interesting files for the user and allows their modification, leaving as output a brand-new executable (keep the original unaltered for reference). This utility does not execute any check whatsoever on what users put in the configuration files, so be careful. The logging system, based on the Apache log4j framework,[35] is intended for java developers, so it is not of great help for the random user.

34 There are three configuration files packed into the jar: `timescape.properties` is a properties file, which configures various aesthetic aspects of the user interface, it can be leaved as-is; `hibernate.cfg.xml` is the database-application link: it must be configured; `log4j.properties` contains the logging parameters.

35 See `https://logging.apache.org/log4j/1.2/` and https://logging.apache.org/log4j/2.x/.

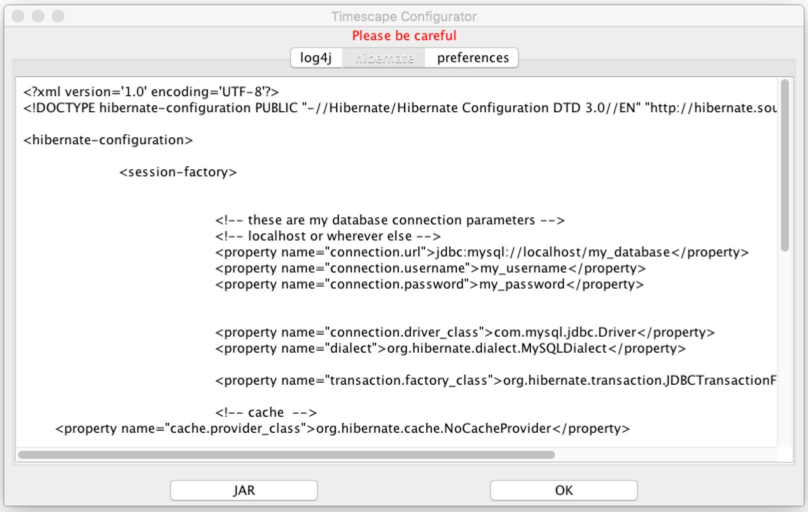


Fig. 2.2 TimescapeConfigurator window

2.2 The Main Window

The main window is just a sequence of buttons, logically ordered following the create-define-evaluate-explore model workflow sketched beforehand. Each button pops a dialog window up for the user to operate on his/her models. At the start, if the database is empty, the program asks for a formatted dataset to parse and store.

Each button corresponds to a logical function, detailed below. Other than the `MANAGER` section, there is a `SETUP` section for defining the model parameters (all fresh models are created the same, with default parameters); a model is then `RUN` in a separate window and, upon completion, it can be examined or explored. From the `EXPLORATION` panel as well the user can export various subsets of the model.

This panel manages the model-level actions which are:

- model creation
- model renaming
- model cloning
- model destruction

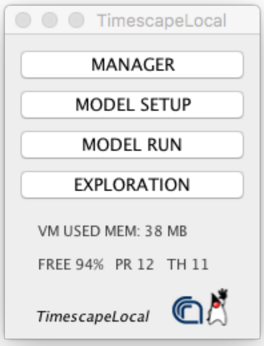


Fig. 2.3

2.3 The MANAGER Panel

The meaning of each operation is self-explanatory. The `CLONE` function is particularly useful to define a collection of models with similar parameters. The `KILL` function operates a clean deletion of the model form the database; care should be taken since there is no "undo" to deletion. The fact that the data are stored on an ordinary database means that they can be accessed straightforwardly within other application. **TimescapeLocal** uses a Hibernate mid-layer for accessing the data, so it can be attached to all supported database flavours.

The example data and the figures that follow are relative to the example included in the distribution. The source dataset is a collection of mycorrhizal δ^{15}N form a symbiosis study.[36]

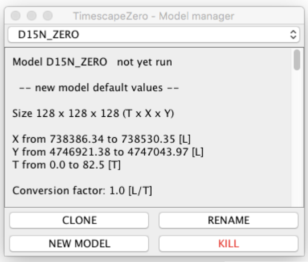
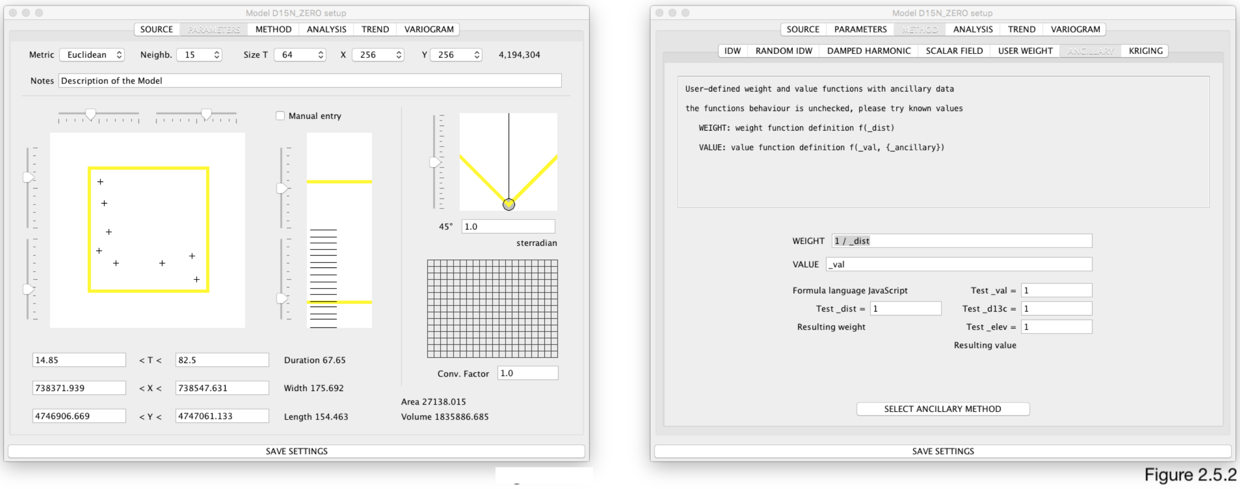


Fig. 2.4

2.4 The SETUP Panel

This section is the most intricate one. Users define their models' parameters through a tabbed dialog window. A `PARAMS` and a `MODEL` panel allow the definition of all the relevant values and of the interpolation method. The parameters that should be defined can be categorised as follows:

36 The δ^{15}N is the relative difference, with respect to an agreed standard, of the ratio of the heavier ^{15}N isotope to the lighter and far more common ^{14}N. It is generally expressed in $^{o}/_{oo}$ units for ease of reading. δ^{15}N is commonly employed in ecophysiology studies.



(a) **(b)**

Fig. 2.5

- model consistency: the number of cells as width (x span) times height (y span) times the number of time sheets,
- boundaries: minimum and maximum bounds for space x and y, and time t coordinates,
- causality: the time to space conversion factor c and the causal cone aperture k,
- neighbourhood: the number of near primes and the metric employed,
- method: the statistical interpolation method to be used.

The `METHOD` panel is subdivided in a set of sub-panels, one for each method implemented. Users can also define their own methods, although this requires non trivial object programming skills. The methods include plain IDW (Inverse Distance Weighted) and some extensions, Kriging, and user-defined functions. Users can also switch off some sample points for advanced statistical testing. Users are assisted by a variety of statistical information about the source dataset:

- The `SOURCE` panel shows all the input dataset details. From this panel the single points can be switched on and off. Ancillary variables values, if present, are also shown here.
- The `ANALYSIS` panel shows a set of statistical analyses about the input dataset, including space and time distribution statistics and correlation analysis.
- the `TREND` panel shows a linear interpolation of the input values vs time and space coordinates (trend analysis). This is particularly useful for the trend removal in the variogram.
- The `VARIOGRAM`, as the name suggests, is the variogram plot. The variogram (and the other statistics) is updated whenever the user changes the value of the c add k parameters, or if a source point is switched.

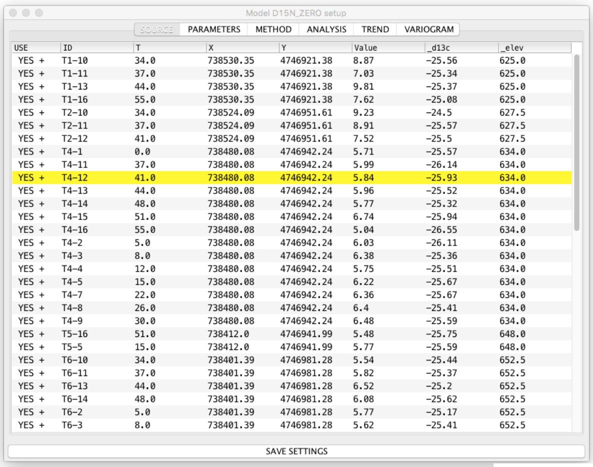
The variogram is the most important piece of information the user is provided with. It is a spacetime variogram: it is computed with the couples of source points which are causally connected according to the value of the c and k parameters.

In the upper right corner of the window there is the number of point couples that affect the variogram; given the causal structure of the spacetime conversion, there is no need to divide the sums by two, as in ordinary variograms. User can control the number of bins displayed and the removal (or not) of the multilinear trend. All single points couples are shown (small black dots) as well as a bin-reduced (thick red crosses) version, showing the general behaviour. The general behaviour is so important that a small window is always open,so that users can see in real time the effect of changing the causal parameters values. The choice of the interpolation method has no effect on the variogram since it computed only with the source points dataset.

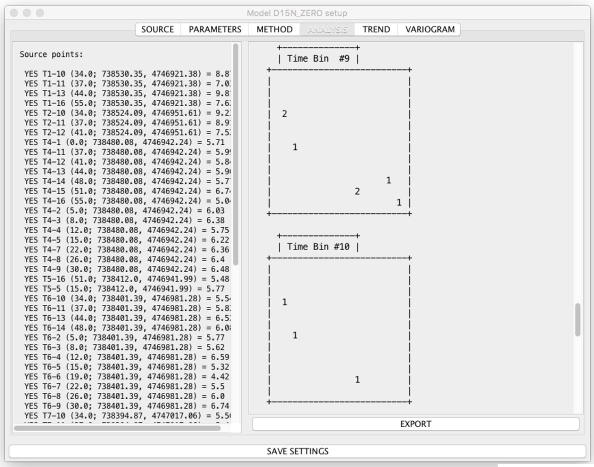
The interpretation of the variogram is eased by the calculation of a few fitting functions: a linear and an harmonic fit,[37] plus the traditional, Kriging-oriented Gaussian, exponential, spherical and double-spherical fits.

The values of the other statistics and trend analyses are not influenced by the c and k parameters, so their update is not so critical; they can change only if a source point is added or removed. The trend over time is especially important: the linear correlation can be negligible $R^2 \approx 0$ but a careful look at the values vs time regression plot can show a periodic component.

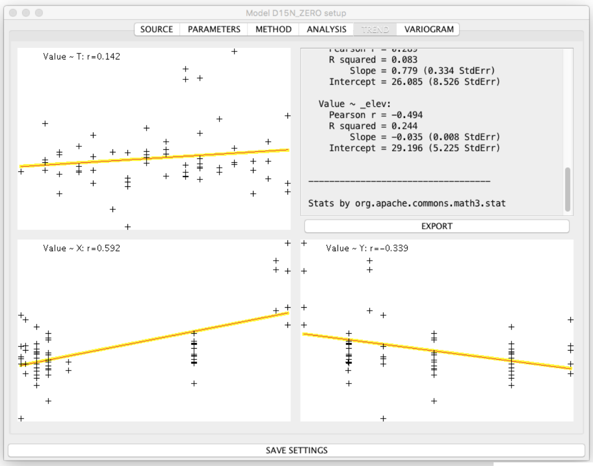
37 An harmonic fit is something unheard of in the realm of ordinary variograms, it reflects the presence of *seasonality*, which induces periodic (harmonic) oscillations of the sampled values. This is a common situation in many ecological studies and in forest ecology in particular, where seasonal variations play a key role.



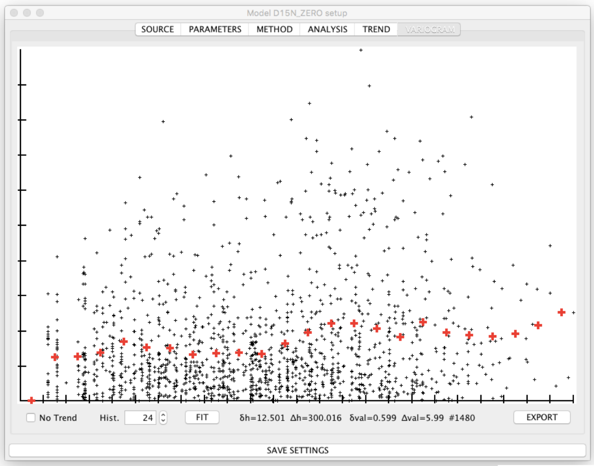
(a)



(b)



(c)



(d)

Fig. 2.6

The IDW method can be adjusted for harmonic components choosing the `DAMPED HARMONIC` interpolator, the period should be tuned according to the pattern of change of the modelled phenomenon, most times it is one year, in forest ecology, where seasonality predominates. Users interested in multi-harmonic components should write themselves the code for the interpolator.

2.5 Model Interpolation

The interpolation function has many customisation opportunities. Building up on the weighted sum idea, the evaluation of the weights is almost free. The most basic option is a simple weighted mean with the weights equal to the inverse of the distance, just a step above, we can consider the rth power of the distance: $w \sim 1/d^r$ maybe with a *mass* m:[38] $w \sim 1/(d^2+m^2)^{\frac{r}{2}}$. All these funcions are pre-packaged in the appropriate tabs of the `METHOD` panel.

An `ANCILLARY` panel allows the user to define his/her interpolation weight functions, complementing the interpolation with ancillary values or oven neglecting the measured values altogether, using the ancillaries alone. An harmonic weight function can be defined as well, as $w \sim \bar{w}(d)\sin(\omega\Delta t + \varphi_0)$, witha a suitable frequency ω and a phase φ_0, $\bar{w}$ is a decreasing function of the distance d.

The interpolation of the model goes on as follows, in a dedicated `RUN MODEL` panel:

- The elements of the model (voxels) are inserted into the database without evaluation (all the values are set to *null*). This step is important for the database safety, since it allows it to grow as fast as possible with a minimum of transaction activities (upper image).
- The voxels are evaluated one after the other, time sheet by time sheet. Each voxel value is then updated on the database.

38 The term *mass* derives from particle physics, where it corresponds to the mass of a scalar field. Here m^2 should be thought of as a dumping factor, a large m depresses the weight, while a small one has little effect on w; the presence of m avoids the blow-up effect of w for the points located close to the source.

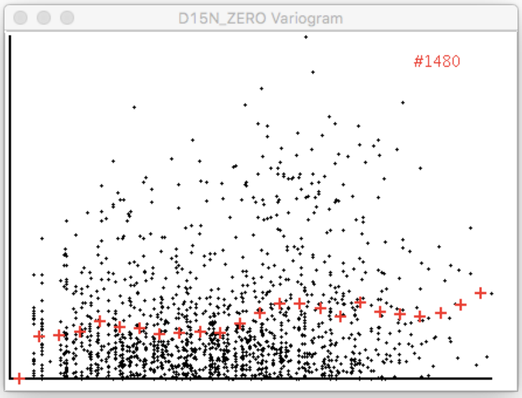


Fig. 2.7 The small variogram window

No table space growing happens in this phase (middle image). The evaluation can be parallelised easily, since there is no relationship among distinct voxels.[39]

- Upon completion, a short report is shown, highlighting any relevant exception[40] that has occurred during the evaluation (lower image).

A progress bar in the lower portion of the panel shows graphically an estimate of the waiting time. Upon completion of each time sheet the user is informed about the time it took for its evaluation. It is important to check from time to time what's going on: an healthy database would take the same time for the insertion of any sheet while, in the evaluation phase, the time intervals between sheets would grow, since the number of *null* voxels shrinks as time goes on.[41]

The evaluation phase, for each voxel v, consists in a normalised summation of the values of the source point which fall inside the set of the possible causes of its central event:

- for each source event s, the distance d from v is $d = \sqrt{c^2(t_v - t_s)^2 + (x_v - x_s)^2 + (y_v - y_s)^2}$ or equivalently $d = \max\{c\,|t_v - t_s|\,, |x_v - x_s|\,, |y_v - y_s|\,\}$
- the associated weight is $w_s = \theta\left(kc\,|t_v - t_s| - \sqrt{(x_v - x_s)^2 + (y_v - y_s)^2}\right) / d$, the Heaviside θ function ensures that an s which is not causally connected with v has a zero weight.
- the associated value f_s is calculated. It can be simply the value of the source point or a more complex function
- the resulting voxel value is the weighted mean of all the values: $V = \left[\sum_s w_s\right]^{-1} \sum_s w_s f_s$. If all the w_s are zero the voxel value remains *null*.

Many variations are possible on the scheme sketched above but all the relevant parameters are controlled in the `SETUP` phase, during the evaluation of the model there is no user interaction, nor it is possible to modify the parameters while a model is running.[42]

The evaluation of a model is particularly core-stressing and the complexity allowed depends on the hardware capabilities. It is always safe to start with downscaled models before running a bulky model. The database activity is intense as well, in the insertion phase the storage space blows up fast but the transactional activity continues all over the run, with as many updates as the model's voxels.

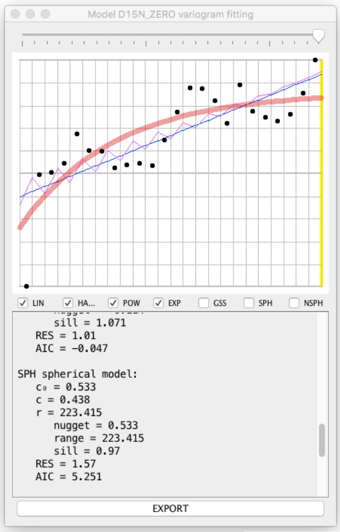


Fig. 2.8

As a rule of thumb, it is advisable to run at least a downscaled model not exceeding $100 \times 100 \times 100$ (one million) voxels before venturing the realm of billions, to see if the hardware is good for the job. Remind that doubling the space and time resolution of a model means an increase of almost $10\times$ of storage space and calculation time.

39 Unlike other kinds of three dimensional models, like the meteorological ones (e.g. weather forecast).
40 Technically, an *exception* does not necessarily mean an error condition.
41 Null voxels are not updated, so no database interaction happens.
42 A running model is in a *locked* state that prevents any modification. If one needs a similar model with different parameters, it is possible to `CLONE` it, also if it is running.

The cell spacing between voxels need not be alike, horizontal and vertical resolutions can be different and the number of time sheet has not to be necessarily comparable with the space resolution.

2.6 The EXPLORATION Panel

A finished model is a collection of voxels in a single database table. A Timescape model does not have a "natural" way of displaying. The information contained in a Timescape has to be extracted somehow, according to the users' needs. The EXPLORATION panel offers a set of statistical and graphical tools for examining the model's values.

The statistical tools consist in three panels (SUMMARY, ANALYSIS and RESIDUALS), while the graphical tools are the most useful ones. It is also possible to export a variety of subsets from a Timescape, to be used as input layers in other GIS or statistical packages. For the sake of storage economy, the actual coordinates (space and time) are not stored in the data records. It is the *id* of the voxels (a triple of integer numbers) that can be used to recalculate the coordinates, so every export filter takes care of rebuilding the coordinates.

The SUMMARY panel shows a synthetic statistical analysis of the model. This report can be exported a a text file or copied to te clipboard for other uses. The informations shown include the geometry of the model (area and volume, both in LLT and LLL units), the number of *null* voxels and its ratio with respect to the total number. There are also some statistics about the values of the model.

The RESIDUALS panel consists in the list of the source point, accompanied by the approximate value of the model in the same location (if it is included in the model extension) and the difference of these, or *residual*. It is also shown a QQ-plot of interpolated vs original values. Also this information can be exported, including the QQ-plot. A little map shows the position of the source points in space and time; clicking on a source point the map and the QQ-plot points are highlighted, to let the user examine in detail the most interesting cases. Lastly, some statistics about the residuals are shown and exported as well.

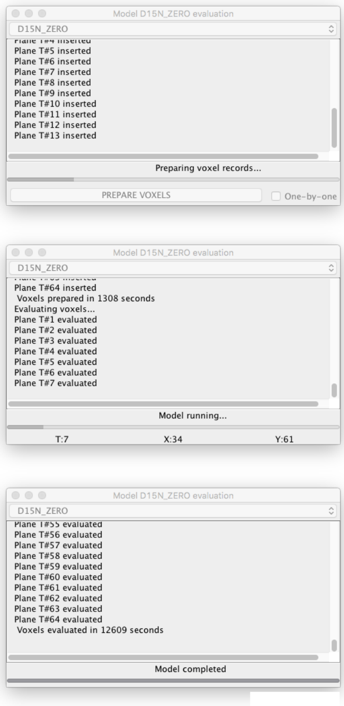


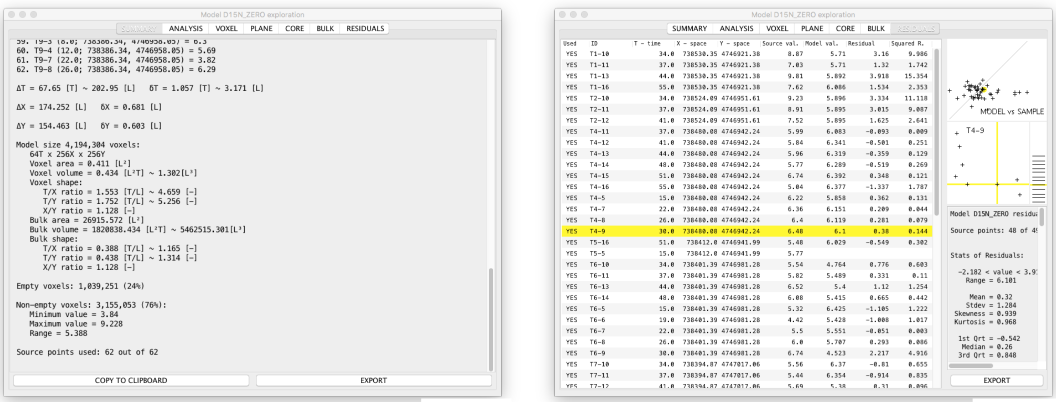
Fig. 2.9

The ANALYSIS panel (below) consists in a sheet-by-sheet analysis of the values of the model. For each time the software calculates the minimum and maximum values, the number of *null* voxels and an histogram, represented as a string of occupation numbers.[43]

Many ecological studies are focused on the variation over time of some quantity. This is the instrument of choice for finding the general behaviour and the extreme changes. Histograms are presented as strings of occupation numbers and not in graphical form due to the lack of space, but one can import such strings in any statistical software whenever a graphical representation is needed.

The most interesting panels are the four that bring a visual representation of some voxels subsets of the model. Basically, the model can be viewed as a cube, the base being the space extension and the height being the time interval. There are many ways in which such a cube can be sectioned, according to the kind of information that the user is looking for.

From a geostatistical point of view, the PLANE panel is the closest possible representation. Each plane of voxel, at a constant



(a) (b)

Fig. 2.10

43 The number of bins is configurable.

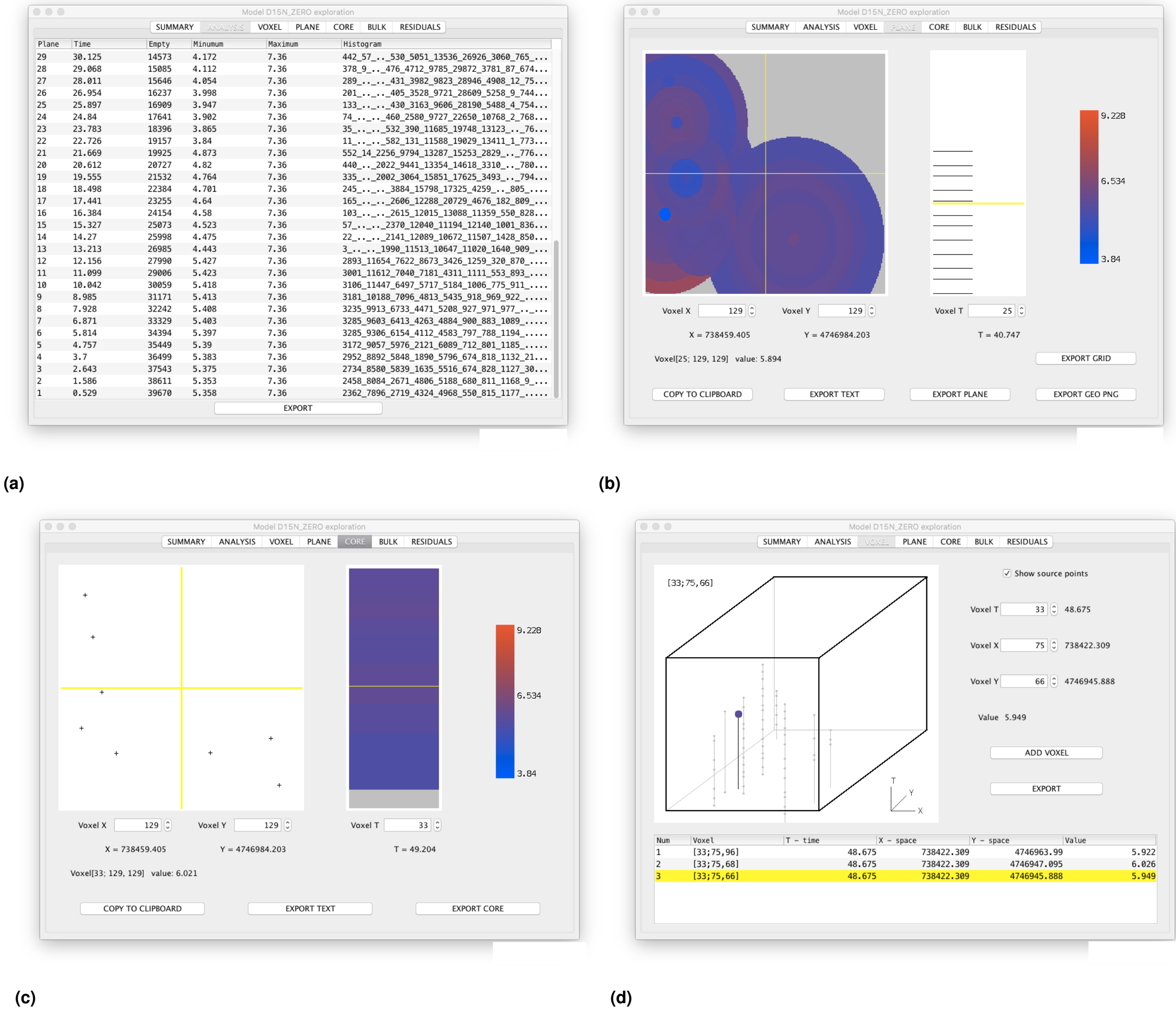


Fig. 2.11

time, is an interpolated surface which can be shipped right to the user's GIS for further analyses. The surface, in fact, can be exported as an ascii grid of georeferenced values,[44] other export format include a georeferenced *.png* rendered image[45] and a formatted text file. The bar in the centre of the panel can be set to the time required; a set of horizontal lines show the time distribution of the samples dataset.

The CORE panel, on the other hand, is devoted to the site-wise analysis of the time evolution of the modelled values. The roles of the left and centre panels is exchanged: the left part shows a map of the area with the location of the samples and and hairline showing the position of the *core* dug into the voxels cube. This core is by any respect a time series of values, which can be exported as well as an ascii collection of values.

There is an endless literature about the treatment of time series in R and other statistical environments. This is where a Timescape can bridge the gap between ordinary "flat" geostatistics and time series analysis. Remember, however, that these time series are not measured values but they are interpolations, or even extrapolations if the maximum model time is greater han the maximum samples time.

The Timescape can be viewed as a cube and explored voxel-by-voxel in the VOXEL panel. This is really for a close inspection of small suspect areas. There is not a global vision of the values as planes or cores, so the user must know what he/she is looking for, and where. Users can build their own table of voxels (the list in the bottom part of the panel) adding them one by one. This table can be exported in ascii format for further statistical investigations.

The main tool for having a sense of what's going on is the BULK tool. Also in this case the model is represented as a floating cube, that can be sliced in all directions.

The model can be sliced horizontally, according constant time surfaces; this is basically what happens in the PLANE tool, too.

44 The format is the standard ESRI GRID.

45 A rendered image is only useful for representation in e.g. Google Earth, but it lack the actual values of the pixels.

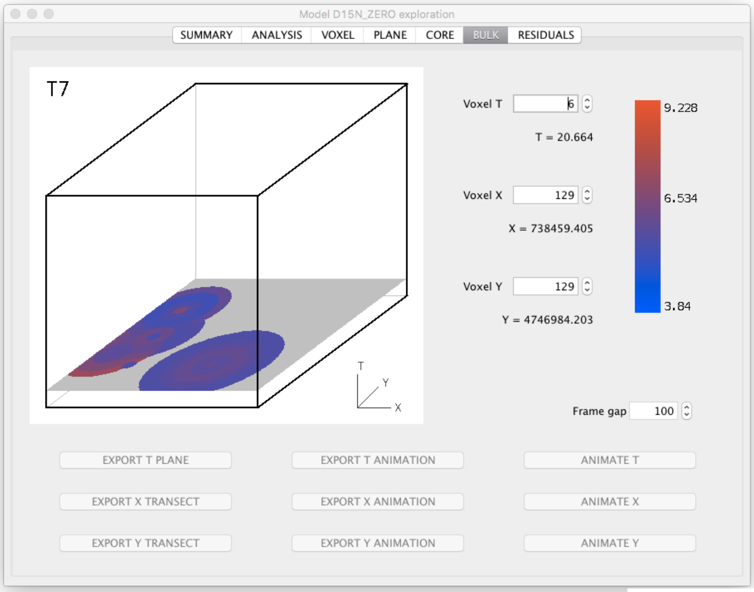


Fig. 2.12 The `BULK` panel

We can also have constant-x and constant-y slices. These represent respectively the time evolution of North-South and East-West transects. A colour bar shows the values of the elements of the model (this legend is also shown in the other panels).

Having a global look at the model is important to find the hot spots and the hot momentum[46] areas, this is what this tool has been designed. This panel is also the first place to see after the completion of a model, since it gives an intuitive graphic representation of the values.

Users can select a value for x, y or t using the input sliders on the right side of the cube, so to have the appropriate plane shown, but it is the animation tool that is the most interesting one. Pressing `ANIMATE X`, `ANIMATE Y` or `ANIMATE T`, the planes are generated in sequence and plot one after the other. It is also possible to export these animations as *.gif* images. The animation procedure is very intensive both in terms of cores load and database accesses (all the elements of the model are selected in sequence), so bottom-of-the-line computers should be used with some care.

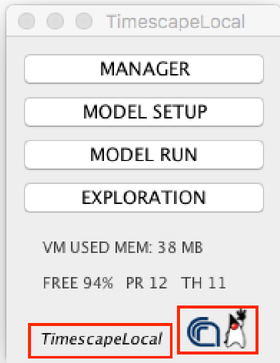


Fig. 2.13

2.7 When Something Goes Wrong

If something has gone bad (say, e.g., if the program has been interrupted by killing the Java Virtual Machine) the database integrity is probably corrupted. If there use has access to the database he/she can try to correct manually the inconsistencies. It is advisable, however, to do so in a more protected way, through a couple of hidden panels that the software offers to the user in trouble. The panels are accessed by double-double-clicking the text label on the lower left corner (single model editing) and the icon on the lower right corner (dataset re-initialisation).

The `HIDDEN MODEL SETUP` panel (left "button") shows up all selected model's characteristics. Everything can be edited and the software performs some rudimentary tests of coherence before actually committing the user's choices.

It is not possible, in any way, to correct the already-evaluated voxels values. It is possible however, unchecking the `Complete` checkbox, to force **TimescapeLocal** to delete any value associated with the model, so to be able to run it again. No modification takes place before explicitly clicking on the `COMMIT` button.[47]

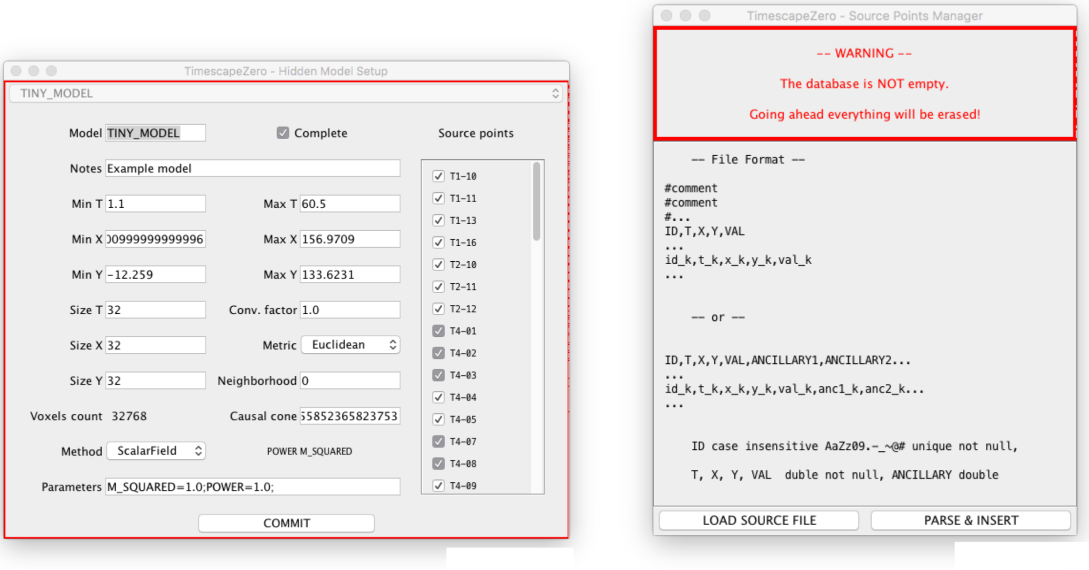
The `SOURCE POINTS MANAGER` (right "button") shows a dialog box where users can put a new source points dataset.

This is the same panel that is opened when **TimescapeLocal** does not find a valid source dataset on the Database, plus a warning message suggesting caution. In fact, providing a new dataset erases everything on the Database. Remember that it is not possible to have different source datasets in the same instance of **TimescapeLocal**: one instance per experiment![48] On the other hand,

46 Hot spots are relatively small areas of distinguishable values, non necessarily related to temperature, while hot momentum refers to small areas of rapidly changing values.

47 No action takes place on the Database before committing, **TimescapeLocal** works on a dummy copy of the model.

48 See section 2.1.3 for configuration details.



(a) (b)

Fig. 2.14

sometimes it is possible to attach ancillary values to the original dataset, but it has to be done before model interpolation, there is no way to append such ancillaries after that the setup of the source dataset has been done.

3 Conclusions

The Timescape algorithm is based on a topological approach which includes causality to define a spatio-temporal distance to be used in any ordinary geostatistical interpolator. Many ecological datasets are georeferenced according to projected (local) coordinates, often the collection strategy includes isolated transacts and scattered points, which do not provide provide a good starting point for geostatistics. The variability of the collection times is also an issue, since the measured variables are almost never stationary. The TimescapeLocal Java application allows the integration of different datasets to produce voxel (i.e. 3D pixel) lattice output that can be included in the users' ordinary GIS workflow.

Timescape Local software is available at the following link: https://sourceforge.net/projects/timescapelocal/