Published in final edited form as:
Neural Comput Appl. 2022 April ; 34(8): 6377–6396. doi:10.1007/s00521-021-06789-8.

Kimesurface Representation and Tensor Linear Modeling of Longitudinal Data

Rongqian Zhang[1,4], **Yupeng Zhang**[2,4], **Yuyao Liu**[1,4], **Yunjie Guo**[3,4], **Yueyang Shen**[3,4], **Daxuan Deng**[3,4], **Yongkai Joshua Qiu**[2], **Ivo D. Dinov**[4,5,*]

[1]Department of Statistics, University of Michigan, Ann Arbor, MI 48109, USA

[2]Department of Mathematics, University of Michigan, Ann Arbor, MI 48109, USA

[3]Electrical Computer Engineering Division, University of Michigan, Ann Arbor, MI 48109, USA

[4]Statistics Online Computational Resource, Department of Health Behavior and Biological Sciences, University of Michigan, Ann Arbor, MI 48109, USA

[5]Department of Computational Medicine and Bioinformatics, University of Michigan, Ann Arbor, MI 48109, USA

Abstract

Many modern techniques for analyzing time-varying longitudinal data rely on parametric models to interrogate the time-courses of univariate or multivariate processes. Typical analytic objectives include utilizing retrospective observations to model current trends, predict prospective trajectories, derive categorical traits, or characterize various relations. Among the many mathematical, statistical, and computational strategies for analyzing longitudinal data, tensor-based linear modeling offers a unique algebraic approach that encodes different characterizations of the observed measurements in terms of state indices.

This paper introduces a new method of representing, modeling, and analyzing repeated-measurement longitudinal data using a generalization of event order from the positive reals to the complex plane. Using complex time (kime), we transform classical time-varying signals as 2D manifolds called kimesurfaces. This kime characterization extends the classical protocols for analyzing time-series data and offers unique opportunities to design novel inference, prediction, classification, and regression techniques based on the corresponding kimesurface manifolds. We define complex time and illustrate alternative time-series to kimesurface transformations. Using the Laplace transform and its inverse, we demonstrate the bijective mapping between time-series and kimesurfaces. A proposed general tensor regression based linear model is validated using functional Magnetic Resonance Imaging (fMRI) data. This kimesurface representation method can

[*]**Contact:** Ivo D. Dinov, Statistics Online Computational Resource, University of Michigan, 426 N. Ingalls Str., Ann Arbor, MI 48109, statistics@umich.edu | URL: https://umich.edu/~dinov | Tel: 1 + 734-764-5556.

Publisher's Disclaimer: This AM is a PDF file of the manuscript accepted for publication after peer review, when applicable, but does not reflect post-acceptance improvements, or any corrections. Use of this AM is subject to the publisher’s embargo period and AM terms of use. Under no circumstances may this AM be shared or distributed under a Creative Commons or other form of open access license, nor may it be reformatted or enhanced, whether by the Author or third parties. See here for Springer Nature’s terms of use for AM versions of subscription articles: https://www.springernature.com/gp/open-research/policies/accepted-manuscript-terms

Conflict of Interest
The authors have no conflicts of interest to declare that are relevant to the content of this article.

be used with a wide range of machine learning algorithms, artificial intelligence tools, analytical approaches, and inferential techniques to interrogate multivariate, complex-domain, and complex-range longitudinal processes.

1. Introduction

Contemporary scientific studies rely on large volumes of multi-source, heterogeneous, and incomplete data anchored at specific spatiotemporal locations. Many state-of-the-art methods for handling time-varying longitudinal data involve parametric models that enable investigation of retrospective, current, and prospective process characterizations. Participant outcomes, experimental conditions (e.g., treatments or exposures), cross-sectional characteristics, and other information is often tracked at multiple points in time. Common strategies for analyzing repeated longitudinal measurements include mixed linear modeling, time-series analysis, growth and decay modeling, forecasting or prospective outcomes, prognostication, causality, survival analysis, and time-to-event inference [1–5].

To account for low signal-to-noise ratio, many study-designs employ repeated data acquisition at controlled experimental conditions. For each unit in the experiment, this leads to assembling multiple samples representing scalar, vector, matrix, or tensor data. In the simplest case, univariate data is collected for a longitudinal process and the goal is to predict the prospective behavior of the process using the observed retrospective (training) data. In more pragmatic settings, the data is represented as a (time) dynamic multi-dimensional array generalizing the classical *samples* × *variables* matrix format. In this paper, we build on the notion of complex time (*kime*) [6] to generalize the classical temporal time-series representation of repeated longitudinal data. Specifically, we model the observed repeated-measurement longitudinal data as 2D kime surfaces. The structure (e.g., geometry and topology) of these higher dimensional manifolds carries more information compared to the original time-series corresponding to cross-sectional kimesurface foliation curves. Model-based methods, such as tensor-based linear modeling [7], and model-free artificial intelligence (AI) approaches, such as unsupervised clustering and classification [3, 8, 9], can then be designed and applied to the kimesurface representations of the time-varying processes.

The main drivers of examining longitudinal data include (1) direct identification of intra-individual change (and stability) and/or of inter-individual differences (similarity); and (2) analysis of interrelationships in behavioral change, causes (determinants) of intra-individual change, and/or of determinants of inter-individual differences [10, 11]. Time-series methods for modeling continuous responses include analysis of response profiles, linear mixed-effects models, hierarchical linear model, and support vector regression [12, 13]. Response profile analysis is typically employed with a relatively small number of repeated measurements, obtained on a common set of occasions [12]. Linear mixed-effects models are generally applicable for data with expected interdependencies, or presence of explicit or implicit hierarchical structure [14]. Mathematical modeling and statistical inference on longitudinal data involves preprocessing to ensure stationary, proper representation of seasonal and non-seasonal effects, accounting for known interventions and outliers, regression estimation and classification analysis. The classical goals of time-series analysis include model fitting, parameter estimation, diagnostic checks, forecasting, and inference [15]. Many model-based approaches rely on autoregressive integrated moving averaging (ARIMA); however there are different classes of techniques for joint modeling of multivariate longitudinal data, e.g., models of evolution of the measured outcomes, marginal models, and models for latent evolution of latent variables [16]. Recent mathematical advances propose using geometric and topological techniques to represent, track, model, and analyze time-series and longitudinal processes [17–19]. Other machine learning and artificial intelligence techniques for time-series analysis include recurrent neural networks, long short-term memory networks, and deep Boltzmann machines [3, 20].

Functional Magnetic Resonance Imaging (fMRI) represents an example of time-varying data that requires careful assessment of underlying model assumptions, method limitations, and data processing challenges associated with longitudinal designs, e.g., task design, sampling strategies, and within subject and cohort-level analyses [13, 21, 22]. Changes in the fMRI time-course corresponds to neuronal activity tracked as variations of the ratio of oxygenated to deoxygenated blood levels, subject to a mediating process known as the Hemodynamic Response Function (HRF). Under various stimuli, e.g., audio, visual, and tactile, a change in the observed fMRI signal is detected in the relevant brain region, subject to HRF mediation with certain response delay, e.g., 2–3 seconds. Magnetic resonance scanners are extremely sensitive to electromagnetic fields and can be used to track brain anatomical structure, tissue spectral composition, and functional network activity. Brain functional activation is a proxy of the level of oxygenated blood hemoglobin molecule, which contains four iron atoms whose magnetization response depends on their oxygen binding; hence oxygenated vs. deoxygenated blood have distinct diamagnetic and paramagnetic properties. As the concentration of oxygenated blood in a brain region reflects changes (increases or decreases) of neuronal activation in response to a specific stimulus, the regional fMRI signal is a proxy indicator highly correlated with the associated brain area processing the specific stimulation task [23].

Tensor-based linear modeling represents longitudinal data using higher-dimensional arrays (tensors) that encode different characterizations of the observed measurements in terms of state indices, e.g.,

$$\underbrace{Conditions \times Repetition}_{\text{experimental design}} \times \underbrace{SpaceX \times SpaceY \times SpaceZ \times Time}_{\text{spatiotemporal characterization}} \times \underbrace{\ldots}_{\text{other}} .$$

For each given tensor element, the observed data may be a (coupled) scalar, vector, matrix, or another tensor. In general, the typical goals of tensor-based linear modeling (TLM) involve obtaining computed class-labels, deriving prospective trajectory forecasts, estimating Bayesian posterior predictive probabilities, and predicting of an outcome tensor $\mathbb{Y}$ of dimension $Q_1 \times \cdots \times Q_M$ from another observed tensor $\mathbb{X}$ of dimension $N \times P_1 \times \cdots \times P_L$, where N is the number of observations [7]. One example of TLM inference is the prediction of several facial attributes from a set of recorded images [24]. In this application, the outcome tensor $\mathbb{Y}: Faces \times Attributes$ is predicted using data in the observed feature tensor $\mathbb{X}: Faces \times X \times Y \times ColorChannels$ [25]. Another application is the prediction of EEG from fMRI data [26]. Typically, EEG has low spatial but high-temporal resolution along the brain cortical surface, whereas fMRI provides better spatial but poorer temporal resolution over the entire 3D spatial domain. TLM allows predicting the association between these two non-invasive functional neuroimaging modalities. Such TLM can be used to enhance derived neuroimaging biomarkers, impute missing or incomplete spatiotemporal information, and derive better prediction for normal and pathological conditions [27, 28].

Appendix A provides some additional background on the tensor definition, representation, and operations. Tensors naturally provide a mechanism for extending unconstrained or regularized linear models for prediction and inference based on high-dimensional and longitudinal data. Consider an observed (training data) outcome $\mathbb{Y}: \mathrm{N} \times \mathrm{Q}_1 \times \cdots \times \mathrm{Q}_\mathrm{M}$ and a corresponding observed covariate tensor $\mathbb{X}: N \times P_1 \times \cdots \times P_L$. Then we can perform (tensor) linear-model fitting:

$$\mathbb{Y} = \langle \mathbb{X}, \mathbb{B} \rangle_L + \mathbb{E} .$$

where the two new tensors are $\mathbb{B}: \mathrm{P}_1 \times \cdots \times \mathrm{P}_\mathrm{L} \times \mathrm{Q}_1 \times \cdots \times \mathrm{Q}_\mathrm{M}$, representing the TLM coefficient (effect-size) tensor, and $\mathbb{E}: N \times Q_1 \times \cdots \times Q_M$, denoting the residual error tensor. Note that the first L modes of $\mathbb{B}$ contract the dimensions of the covariate tensor, $\mathbb{X}$, which are not in the outcome tensor $\mathbb{Y}$, and the last M modes of $\mathbb{B}$ expand along the outcome $\mathbb{Y}$ tensor modes in $\mathbb{Y}$ not present in the covariate tensor $\mathbb{X}$. Assuming we have sufficient data to fit this model and estimate the effect tensor $\mathbb{B}$ and the residual error tensor $\mathbb{E}$, then we can perform prediction and forecasting of the outcome tensor using classical linear modeling strategies. For a given outcome tensor index-set $(q_1, \cdots, q_M)$ the TLM prediction of the outcome tensor element for each case $1 \leq n \leq N$ is:

$$\mathbb{Y}[n, q_1, \cdots, q_M] \cong \sum_{p_1 = 1}^{P_1} \cdots \sum_{p_L = 1}^{P_L} \mathbb{X}[n, p_1, \cdots, p_L] \mathbb{B}[p_1, \cdots, p_L; q_1, \cdots, q_M] .$$

To simplify the notation, the TLM model excludes the static tensor-intercept term and assumes that the observed data tensors $\mathbb{Y}$ and $\mathbb{X}$ are centralized.

This paper is focused on representing, modeling, and analyzing repeated-measurement longitudinal data as complex time (kime) surfaces. This characterization of time-varying information facilitates the analysis of classical time-series as higher-dimensional kimesurface manifolds. The manuscript is organized as follows. In Section 2, we define complex time and present alternative time-series to kimesurface transformations. The following Section 3 describes the general tensor regression based linear modeling approach. Section 4 includes applications of spacekime TLM analytics to functional Magnetic Resonance Imaging (fMRI). Finally, in Section 5, we summarize the spacekime analytics approach and interpret the reported findings. The supplementary materials section (Appendix) contains specific implementation details and additional examples.

2. Complex-time (kime) and Spacekime Representation

Complex time (*kime*) extends the familiar notion of event order (time) to the complex plane. The kime representation of longitudinal data involves indexing measurements by $\kappa = te^{i\theta}$, where t and θ represent the longitudinal event order (time) and a directional kime-phase (repetition index), respectively. The kime-phase θ is indirectly related to the classical repeated random (independent and identically distributed) sampling during the process of recurrent measurement. The rationale for this extension of time from the positive reals ($\mathbb{R}$) to the complex plane ($\mathbb{C}$) is three-fold. First, in a classical mathematical sense, the longitudinal axis of spacetime represents only a positive cone over the field of the real numbers [29]. The time dimension forms a subgroup of the multiplicative group of the reals, however it's not closed under addition. On the other hand, complex-time describes an algebraic prime field [30] that naturally extends the reals and hence, the classical temporal domain. Although time is ordered, whereas kime is not, the intrinsic time ordering is preserved in the kime magnitude. Second, the kime dimensionality lift addresses some of the physical problems of time [31]. Third, repeated-measurement longitudinal data are represented in terms of kime as 2D manifolds. This enables a higher-dimensional view of longitudinal processes and offers unique opportunities to design and validate innovative data-driven artificial intelligence (AI) and statistical approaches for modeling, analysis, inference, and prediction on temporal signals [6].

Although there are multiple strategies to transform longitudinal processes to kimesurfaces, we will demonstrate this transformation using two specific kimesurface representation methods [6]. Both methods can translate observed time-varying signals, such as spacetime fMRI data, into spacekime kimesurfaces. The first *empirical strategy* is based on discretely estimating the "missing/unobserved" kime-phases and directly mapping the observed longitudinal-indexed time-series into kime-indexed surfaces. The second *analytical strategy* for mapping between time-domain data and kime-domain data utilizes the Laplace Transformation [32].

For more specificity, we will explicate both time-series to kimesurface transformation strategies in terms of a finger-tapping fMRI experiment. In the empirical strategy, the longitudinal (time) indexing, $x(t)$, includes one epoch of the fMRI design. The second component, kime-phase indexing, $x(\varphi)$, runs over the number of epochs of one specific stimulus (e.g., finger-tapping or rest). These phase directions are generally unknown and may be randomly drawn (or estimated) from the appropriate kime-phase distribution for the specific stimulus condition. Figure 1 shows a direct mapping between fMRI time-series and their corresponding kimesurface counterparts. The entire fMRI time course is spliced and arranged according to the epoch curves into an approximate piecewise parametric surface indexed over kime with complex-time intensities derived from the corresponding complex-valued fMRI signal. The complex-value of the function $f(t, \varphi) = Ae^{i\theta}$ is in the range-space and has magnitude $A = A(t, \varphi)$ and (range) phase $\theta = \theta(t, \varphi)$ that correspond to the polar coordinate representation of complex time (kime), i.e., (t, φ).

Figure 2 demonstrates two views of the same 3D scenes depicting an empirical kimesurface reconstruction of the *on* (stimulus) vs. *off* (rest) fMRI signal. The graphs show the kime domain areas where the on vs. off differences are small (surface valleys) or large (extreme high or low surface peaks). The former indicate kime ranges where the voxel location corresponding to the kimesurface does not exhibit significant brain activation. Whereas the latter, surface peak regions, correspond to kime areas of potentially statistically significant on vs. off brain activation associated with the cognitive stimuli. Note that the practical and statistical significance of these peaks and valleys can be evaluated using Gaussian random field theory of the excursion sets above certain threshold values [33, 34].

To map longitudinal time-series into kimesurfaces using an alternative *analytical strategy*, we can employ the Laplace Transform (LT). LT converts complex-valued functions of positive real variables (e.g., time) to complex-valued functions defined on complex variables (e.g., kime). In general, for a function $f(t):\mathbb{R}^+ \to \mathbb{C}$, the continuous LT is defined by:

$$F(z) \equiv \mathscr{L}(f)(z) = \int_0^\infty f(t)e^{-zt}dt:\mathbb{C} \to \mathbb{C}\,.$$

In our situation, complex-valued fMRI time-series signal $f(t):\mathbb{R}^+ \to \mathbb{C}$ can be represented as an ordered (discrete) sequence of complex-valued fMRI BOLD intensities:

$$\{f(t) = a_t + ib_t : 1 \le t \le T\}\,.$$

The discrete LT of f is represented as an infinite series corresponding to the discretized LT integral above. For a specified step-size $\eta > 0$, the discrete Laplace transform $\mathscr{L}(f)$ is computed by:

$$\mathscr{L}_\eta(f)(z) = \eta \sum_{k=0}^{\infty} e^{-zk\eta} f(k\eta)\,.$$

To demonstrate the Laplace Transform-based analytical strategy, we apply the Inverse Laplace Transform (ILT) on an fMRI kimesurface and then convert it back to the original fMRI time-series, see Figure 3. Given a complex function $F(z)$, which represents a kimesurface, and a positive real argument t, the ILT algorithm, developed by Valsa and Brancik [35], involves iterative continuous approximations to generate an approximation of the output $f(t) = \mathscr{L}^{-1}(F)(t)$. This iterative approximation approach estimates the exponential term e^{-zt} using a series where each term is a Bromwich integral obtained through residue calculus [36]:

$$f(t) = \mathscr{L}^{-1}(F)(t) = \frac{1}{2\pi i}\lim_{T\to\infty}\int_{\gamma - iT}^{\gamma + iT} e^{zt}F(z)dz = f_c(t,a) + O\left(e^{-2a}\right)$$
$$= \frac{e^a}{2i}\int_{\gamma - i\infty}^{\gamma + i\infty}\Bigg(F(z)\sum_{n=0}^{\infty}\Bigg[\frac{(-1)^n\left(n+\frac{1}{2}\right)}{\left(n+\frac{1}{2}\right)^2\pi^2 + (a - zt)^2}\Bigg]\Bigg]\Bigg)ds + O\left(e^{-2a}\right) = -\frac{\mathrm{e}^a}{t}\sum_{n=0}^{\infty}(-1)^n\,\mathrm{Im}\left\{F\left[\frac{a}{t} + \mathrm{j}\left(n+\frac{1}{2}\right)\frac{\pi}{t}\right]\right\}$$
$$+ O\left(e^{-2a}\right),$$

where a > 0 is a parameter that can be chosen arbitrarily large to control the error rate, and the subscript c stands for cosh(·), which is involved in the intermediate steps. In practice, the Euler transformation [37] may be used to accelerate the convergence of the alternating series above.

3. Methods

3.1 Datasets

We tested the process of mapping observed time-series data to kimesurfaces, reconstruction of the time-series process, the subsequent analytical modeling, and statistical inference using real Functional Magnetic Resonance imaging (fMRI) data.

Specifically, we describe the results of a neuroimaging study using an “on-off” fMRI finger-tapping experiment, which models the brain response via an oscillatory step-wise characteristic function with a delayed correlation to the on-off stimulus paradigm. The observed data represents complex-valued fMRI time-series of the blood oxygenation dependent (BOLD) response to a dichotomous finger-tapping “on vs. off” (activation vs. rest) event-related experiment [38, 39]. The raw signal is acquired in the Fourier k-space over a period of 8-minutes tracking the functional BOLD signal of a normal volunteer during the finger-tapping “on-off” task. Using the inverse Fourier transform, the data was mapped into spacetime and stored as a 4D double-precision array of complex-valued intensities. The size of the data is about ½ GB (gigabytes) and the dimensions of the data are $\underbrace{64x \times 64y \times 40z}_{\text{3D spatial voxel, } v} \times \underbrace{160t}_{\text{time}}$. Each epoch of the longitudinal (time) finger-tapping task represents a basic pattern of 10 ON (activation) time-points followed by 10 OFF (rest) time-points. The ON and OFF epochs are intertwined and repeated 8 times for a total of 160 time-points, each of about 3 seconds. We applied an established fMRI data preprocessing protocol, e.g., using the R package *fmri*, to motion-correct, skull-strip, register, and normalize the raw fMRI signal [40, 41]. The target of the registration was the LONI Probabilistic brain atlas [42], which allows anatomically parcellating the functional brain data into 28 left and 28 right hemisphere cortical regions. This last step was important for the subsequent three-phase statistical inference aiming to identify the spatial locations in the brain, $\{\nu = (x, y, z)\}$, associated with the finger-tapping motor task.

3.2 Tensor-based linear modeling

In order to detect brain activated areas associated with the finger-tapping somatosensory motor task, we apply tensor-based linear modeling to the spacekime represented fMRI data. Our analytical protocol includes *three phases*. In *phase 1*, classical statistical methods are used to raw fMRI data in order to identify any Regions of Interest (ROIs) that may be activated by the external stimulus. In *phase 2*, we apply tensor regression to the corresponding kimesurface representations of the fMRI data. And finally, to reduce the false-positive rate, in *phase 3*, we apply post-hoc correction to the p-values computed in phase 2.

In the preprocessing step, the real- or complex-valued fMRI data can be co-registered or aligned with a specific brain atlas using linear or nonlinear transformation techniques [42–

44]. Then, we continued partitioning the brain anatomy into Regions of Interest (ROIs) by using the probabilistic atlas, which assigns likelihoods to be inside any 56 ROIs [42]. In this way, each voxel is effectively tagged with a label representing its most likely ROI anatomical localization. In this case, we used the LONI probabilistic brain atlas to tessellate the entire brain anatomical space into 56 complementary ROIs.

The aim of *phase 1* analysis is to select important ROI candidates that can subsequently be interrogated voxel-by-voxel to further localize the statistical significance maps reflecting the activated brain areas associated with the finger-tapping task. We used a measure called temporal Contrast-to-noise Ratio (tCNR) [45] to represent the average signal-change or task-related variability relative to the non-task-related variability over time. In fMRI studies, two strategies are commonly used the estimate the tCNR [46]. The first one is based ion modeling the known stimulus design and utilizing the hemodynamic response function. The second strategy involves low-pass filtering and assumes only that most high-frequency fMRI components contain temporal noise. For a fixed ROI, the most general form of the tCNR is represented by:

$$\mathrm{tCNR} = \frac{\Delta S}{\sigma_{noise}^{t}},$$

where the numerator ΔS is the average signal-change reflecting task-related variability and the denominator σ_{noise}^{t} represents the non-task-related variability over time. Strong tCNR evidence of a task-specific effect within an ROI is indicated by tCNR $\gg 1$, whereas tCNR ≈ 1 suggests the regional may not be involved in stimulus processing. We compute an estimate of the true fMRI signal at a given voxel location ν by using polynomial regression, low-pass filtering, or a convolution with a smoothing kernel. This smoother fMRI signal is used to compute the numerator, ΔS_{ν}, and the variability of the time-series noise, $\sigma_{\upsilon,\,noise}^{t}$, is estimated as the difference between the observed and smoothed fMRI signals. The overall ROI tCNR is computed by pooling the local estimates, $\mathrm{tCNR}_{\upsilon} = \frac{\Delta S_{\upsilon}}{\sigma_{\upsilon,\,noise}^{t}}$, over all voxels $\nu \in$ ROI.

Parametric or non-parametric statistical tests, such as t-test and Wilcoxon test [47], can be used to quantify the ROI-wise stimulus related to brain activation. In our case, we used a *t*-test to assess the statistical significance in each of the 56 ROIs independently. The test-statistics can be computed using the observed tCNR vector $(tCNR_1, \mathrm{t}CNR_2, \cdots, \mathrm{t}CNR_N)^T$, where N is the number of voxels in the ROI. To control the type-I (false-positive) error rate, we used Bonferroni correction for multiple testing to adjust the resulting ROI p-values (α=0.000892).

In *phase 2*, the tensor linear regression models were estimated separately on each of the 56 ROIs to obtain efficient model parameter estimates. For each (irregularly shaped) ROI, we first encapsulated it with the smallest bounding box. This facilitates the tensor arithmetic by padding the fMRI intensities with zeros outside of the ROI boundaries. We can denote the general dimensions of the smallest bounding box for each ROI by $a \times b \times c$. Suppose

$\mathbb{Y}: t \times a \times b \times c$ is the response tensor, i.e., fMRI intensity and $\mathbb{X}: t \times p \times 1$ is the predictor tensor. Thus, the proposed tensor-based linear model is given by:

$$\mathbb{Y} = \underbrace{\langle \mathbb{X}, \mathbb{B} \rangle_L}_{\substack{\text{tensor} \\ \text{product}}} + \mathbb{E},$$

where $\mathbb{B}: p \times 1 \times a \times b \times c$ is a tensor linear model coefficient and $\mathbb{E}: t \times a \times b \times c$ is the residual error tensor.

The design tensor $\mathbb{X}$ contains the information corresponding to the on-off stimulus, i.e., finger-tapping task indicator $\{+1(On), -1(Off)\}$, adjusted by the Hemodynamic Response Function (HRF) which models the expected blood oxygen level dependent (BOLD) response. In addition, $\mathbb{X}$ includes a pair of polynomial drift terms; one of order one (linear) and the other of order two (quadratic) to adjust for low-frequency noise.

We fitted this model by using the R package *MultiwayRegression* [25], which estimates coefficients using penalized least-squares. Then, parametric or non-parametric statistical tests can be applied to the estimated BOLD coefficients $\hat{\beta}$ to enable statistical inference. In this case, we used *t*-tests on the estimated coefficients $\hat{\beta}$ in each ROI. The smaller the corresponding p-values are, the more significant the neuro-activation is in the respective brain area.

Phase 3 of the protocol involves post-hoc analysis, which is necessary for filtering potential false positive discoveries and only exposing true-positive brain activation regions. In this final phase of the analysis protocol, we used a two-fold strategy. First, we applied a false discovery rate (FDR) correction to account for multiple testing [48], and second, we used a spatial clustering filter to control for the size of the activation region [49]. The pair of post-hoc correction strategies used in this phase are designed to temper erratic noise (FDR), as well as, to annihilate smaller-in-size regions of activated voxels (clustering), which are more likely to represent sporadic random activations.

4. Applications

To compare classical space-time analytics vs. spacekime inference, we applied this three-phase tensor-based analytical protocol to the original *spacetime* fMRI finger-tapping task, as well as to its corresponding *spacekime* representation. Then we contrasted the statistical significance maps, i.e., computed and compared the p-values corresponding to the on-off stimulus. The results were visualized in terms of the ROI cluster size and the p-values corresponding to brain activation blocks within the specific ROIs, see Figure 4.

The results on Figure 4 show some important advantages of our three-phase tensor-based analytical protocol. *First*, unlike traditional voxel-by-voxel statistical inference on fMRI data, our method takes spatial correlation into consideration by using tensor regression, which is more in line with the fact that the signal at a particular voxel is likely to be similar to the signals of nearby neighboring voxels. *Second*, this method substantially reduces the dimensionality of fMRI data, which leads to efficient estimation and prediction. In addition, the comparison of the results obtained using the time-series and kimesurface analyses clearly indicates that spacekime represented fMRI data can retain more information than the raw fMRI time-courses. This suggests that kimesurface representation and spacekime analytics may provide more reliable activation brain maps. The activated brain areas revealed by the tensor-based statistical analysis of the kimesurfaces associated with the finger-tapping task paradigm are consistent with the evidence of observing such activation in the somatosensory cortex (motor area), which validates the performance of the spacekime analytical strategy for modeling, inference and prediction based on fMRI signals. Appendix B provides some additional renditions of 3D scenes showing the results of each phase of the 3-tier spacekime analytical protocol.

5. Conclusions and Discussion

Introducing complex-time provides a foundation to address many interesting scientific challenges, including proposing novel techniques for interrogating time-varying longitudinal data, addressing some of the problems of time, and extending and generalizing mathematical equations describing natural laws of physics. The kime representation of longitudinal observations lifts the familiar notion of event order (time) to the complex plane. The induced spacekime representation expands the longitudinal dimension of time and generalizes the classical 4D universal spacetime to a 5D spacekime manifold whose topological structure facilitates novel data analytics. Spacekime representation of data can be used to design, test, and validate novel statistical models for risk estimation, probabilistic modeling, trajectory forecasting, parametric and non-parametric inference, as well as supervised and unsupervised artificial intelligence.

In this paper, we proposed discrete and continuous kimesurface representation strategies for transforming longitudinal data into kimesurfaces. Applying general tensor regression to the resulting kimesurfaces allows the comparison and contrasting of classical time-series analysis with tensor regression on kimesurfaces. Using fMRI data, we presented a three-phase tensor-based analytical protocol for the spacetime and spacekime represented data. We showed that this protocol substantially reduces the dimensionality of fMRI data, which leads to efficient estimation and prediction. We also showed that spacekime fMRI data representation can retain more information than spacetime represented fMRI data, thus revealing more activated brain areas. Therefore, one advantage of kimesurface is its ability to potentially expose supplemental information that may enhance traditional spacetime observation-based scientific inference, improve data-driven predictions, and refine evidence-based decision-making processes.

The main contribution of this article revolves around the introduction of a tensor-based linear model applicable for representation, analysis, and inference of repeated-measure longitudinal data following a transformation from the linear (positive real) time domain to the (complex) kime plane.

Two alternative strategies for mapping time-series data to kimesurface manifolds are presented and tested on neuroimaging fMRI data. This spacekime analytical technique offers a new mechanism to represent time-varying and high-dimensional datasets and provides a foundation for developing innovative ML/AI methods that capitalize on the intrinsic topological structure of the spacekime data manifolds. In addition to applications in biomedical sciences [50], econometrics [51], and environmental sciences [52], the spacekime representation and tensor analytics may be useful in a number of biophysics studies [53], thermoelastic diffusion solutions [54], and boundary value problems [55].

As firm supporters of open-science, we have documented, packaged and released the source-code in a time complexity and inferential uncertainty (TCIU) R-package. The source code, the TCIU package, vignettes and documentation are available on GitHub (https://github.com/SOCR/TCIU) and CRAN (https://cran.r-project.org/web/packages/TCIU/). More information, interactive demonstrations, and additional materials are available on the spacekime website (https://spacekime.org). Community input is always welcome.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

This research is supported in part by funding provided by the National Science Foundation, the National Institutes of Health, and the Michigan Institute for Data Science. National Institute of Health (https://www.nih.gov) grants: UL1 TR002240, R01 CA233487, R01 MH121079, T32 GM141746; National Science Foundation (https://www.nsf.gov) grants: 1916425, 1734853, 1636840, 1416953, 0716055, 1023115. The funders had no role in the study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Appendix

Appendix A.: Introduction to Tensors

A tensor is an important mathematical object defined as a multilinear map from a vector space (module) over a fixed field (or ring) into the field (or ring). In the special case where the field is $\mathbb{R}$ or $\mathbb{C}$ and the vector space is finite dimensional, these multilinear maps coincide with the common computational and data science tensor definition as multidimensional arrays. The space of multidimensional arrays is then naturally isomorphic to the space of multilinear maps. In physical sciences, a tensor maps a set of K vector-space elements and M dual-space elements into the base field, $\mathbb{F}$.

$$\underbrace{\text{Tensor as multilinear map}}_{\text{abstract tensor product}} : \underbrace{V_1 \times \cdots \times V_K}_{K\text{ elements}} \to \mathbb{F}, \underbrace{\text{Tensor as multiway array}}_{\text{data science}} : \mathbb{F}^{I_1 \times I_2 \times \cdots \times I_K}$$
$$\underbrace{\text{Tensor}}_{\text{physics}} : \underbrace{V_1^* \times \cdots \times V_K^*}_{\substack{M\text{ elements}\\ \text{in dual space}}} \times \underbrace{V_1 \times \cdots \times V_K}_{\substack{K\text{ elements in}\\ \text{vector space}}} \to \mathbb{F},$$

where $\mathbb{F}$ can be either $\mathbb{R}$ or $\mathbb{C}$, V_i is a vector space over $\mathbb{F}$, and V_i^* is its dual, $\forall\ 1 \le i \le K$. Throughout this paper, we refer to the data science definition of tensors as multiway arrays representing computable data objects tracking observed collections of multivariate features organized into K-way arrays. Common TLM notation uses lowercase bold symbols

($\boldsymbol{a}$) for vectors, bold uppercase ($\boldsymbol{A}$) for matrices, and uppercase blackboard symbols ($\mathbb{B}$) as tensors of any dimension. Note that the base fields are also denoted by blackboard symbols ($\mathbb{F}$). A K^{th} order tensor is a multidimensional array $\mathbb{B}: I_1 \times \cdots \times I_K$ with I_k representing the dimension of the k^{th} *mode* of the tensor, $\forall 1 \le k \le K$. The tensor elements are specified as $\mathbb{B}[i_1, \cdots, i_K]$, where $1 \le i_k \le I_K$, $\forall 1 \le k \le K$. Computable data tensor objects can be represented as outer products, °, of a set of vectors $\{\boldsymbol{b}_k\}_{k=1}^{K}$:

$$\mathbb{B} = \boldsymbol{b}_1 \circ \boldsymbol{b}_2 \circ \cdots \circ \boldsymbol{b}_K,$$

with its elements indexed as:

$$\mathbb{B}[i_1, \cdots, i_K] = \prod_{k=1}^{K} \boldsymbol{b}_k[i_k].$$

The outer product operation ° takes a pair of a K_1^{th}-order and a K_2^{th}-order tensors as inputs, and returns a $(K_1 + K_2)^{th}$-order tensor. In general, transformation tensors can be considered as multilinear mappings between vector spaces. If $\mathbb{F}$ denotes the real ($\mathbb{R}$) or complex ($\mathbb{C}$) field including the tensor elements, then a data tensor is an object that can be represented as

$$\mathbb{F}^{I_1 \times I_2 \times \cdots \times I_K} \cong \underbrace{\mathbb{F}^{I_1} \otimes \mathbb{F}^{I_2} \otimes \cdots \otimes \mathbb{F}^{I_K}}_{\text{vector space tensor product}} \cong \underbrace{\mathscr{L}}_{\text{multilinear map}} \left(\mathbb{F}^{I_1} \times \mathbb{F}^{I_2} \times \cdots \times \mathbb{F}^{I_K}, \mathbb{F}\right).$$

The *order* of a given tensor, K, represents the number of tensor modes. For each tensor, there exists an integer R (min R is the tensor *rank*) that allows an expansion of the tensor as a linear combination of R rank-1 tensors:

$$vec(\mathbb{B}) = \sum_{r=1}^{R} \lambda_r(\boldsymbol{b}_{K,r} \overbrace{\otimes \boldsymbol{b}_{K-1,r} \cdots \otimes \boldsymbol{b}_{k,r} \otimes \cdots \otimes}^{\text{Kronecker tensor product}} \boldsymbol{b}_{1,r}),$$

$$\mathbb{B} = \sum_{r=1}^{R} \lambda_r \boldsymbol{b}_{1,r} \circ \boldsymbol{b}_{2,r} \circ \cdots \circ \boldsymbol{b}_{k,r} \circ \cdots \circ \boldsymbol{b}_{K,r},$$

where $\lambda_r \in F$, $\boldsymbol{b}_{k,r} \in \mathbb{F}^{I_k}$, and $1 \le k \le K$. The minimal min R that permits such representation is called the *rank* of the tensor and the associated tensor decomposition is called minimal CANDECOMP/PARAFAC (CP) or Polyadic tensor decomposition [7, 56].

The tensor vectorization operation $vec(\mathbb{B})$ restructures (or vectorizes) the tensor as a 1D vector containing all tensor array entries. The length of the corresponding $vec(\mathbb{B})$ is $\prod_{k=1}^{K} I_k$ and the indexing transformation is defined by:

$$vec(\mathbb{B})\left[i_1 + \sum_{k=2}^{K}\left\{\left(\prod_{l=1}^{k-1} I_l\right)(i_k - 1)\right\}\right] = \mathbb{B}[i_i, i_2, \cdots, i_K].$$

Tensors can also be unwounded as matrices by unfolding them along a specified tensor mode k. The mode-k matrix representation of the tensor $\mathbb{B}$ is $\boldsymbol{B}^{(k)}: I_k \times \prod_{l \neq k} I_l$ and contains the vectorized representation of each sub-tensor in the k^{th} mode.

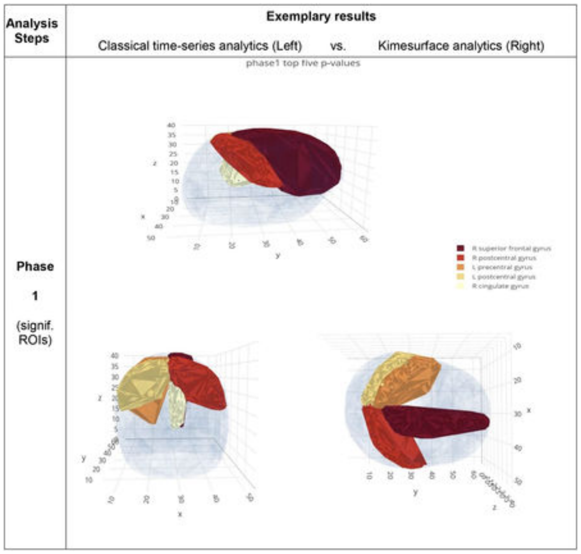
Finally, for a pair of tensors $\mathbb{A}$ and $\mathbb{B}$, we can define the *contracted tensor product*, $\langle \mathbb{A}, \mathbb{B} \rangle_L$, which naturally leads to tensor-based linear modeling, tensor inference, and linear tensor model prediction.

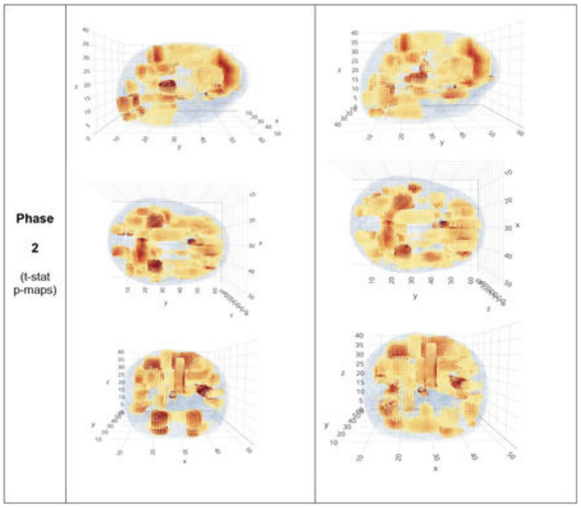
$$\mathbb{A}: I_1 \times \cdots \times I_K \times P_1 \times \cdots \times P_L \quad \text{and} \quad \mathbb{B}: P_1 \times \cdots \times P_L \times Q_1 \times \cdots \times Q_M, \langle \mathbb{A}, \mathbb{B} \rangle_L: I_1 \times \cdots \times I_K \times Q_1 \times \cdots \times Q_M, \langle \mathbb{A}, \mathbb{B} \rangle_L[i_1, \cdots, i_K; q_1, \cdots, q_M] = \sum_{p_1=1}^{P_1} \cdots \sum_{p_L=1}^{P_L} \mathbb{A}[i_1, \cdots, i_K; p_1, \cdots, p_L]\mathbb{B}[p_1, \cdots, p_L; q_1, \cdots, q_M].$$

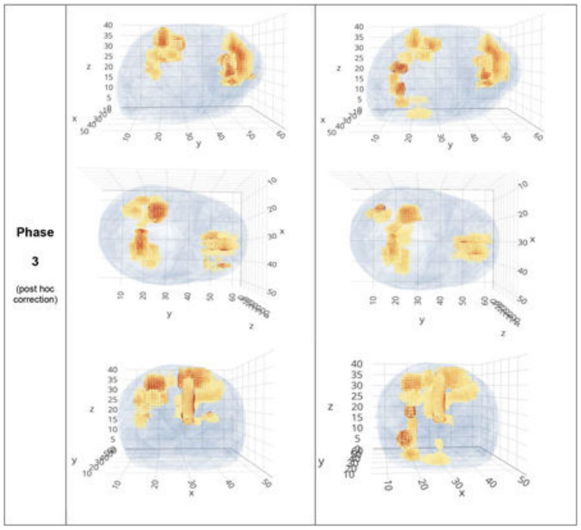
Notice that in this example the contracted tensor product takes an order-($K + L$) tensor and an order-($L + M$) and returns a ($K + M$)-tensor, as opposed to ($K + 2L + M$)-tensor with an outer product. Specifically, the tensor product contraction clears the common dimensions shared by the two tensors.

Appendix B.: Additional 3D Views of the Results

The composite figure below extends the results shown in Figure 4 of the paper by providing additional 3D views of different scenes of the results of each of the proposed 3-phase spacekime analytical protocol.

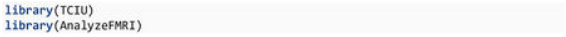






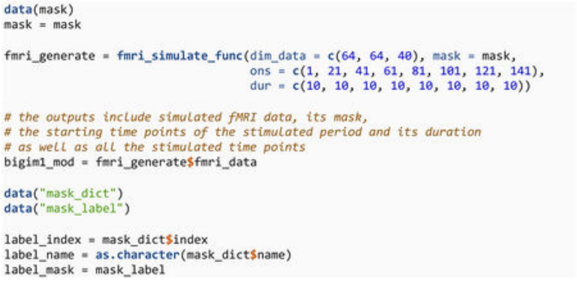
Appendix C.: Implementation of Tensor-based linear modeling

I. Time Complexity and Inferential Uncertainty (TCIU) R-Package Loading



```
library(TCIU)
library(AnalyzeFMRI)
```

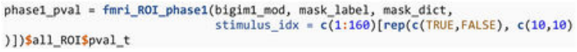
II. Data Manipulation



```
data(mask)
mask = mask

fmri_generate = fmri_simulate_func(dim_data = c(64, 64, 40), mask = mask,
                                   ons = c(1, 21, 41, 61, 81, 101, 121, 141),
                                   dur = c(10, 10, 10, 10, 10, 10, 10, 10))

# the outputs include simulated fMRI data, its mask,
# the starting time points of the stimulated period and its duration
# as well as all the stimulated time points
bigim1_mod = fmri_generate$fmri_data

data("mask_dict")
data("mask_label")

label_index = mask_dict$index
label_name = as.character(mask_dict$name)
label_mask = mask_label
```

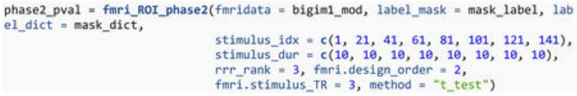
III. Three-phase ROI Analysis

a. Phase1: Detect Potential Activated ROI



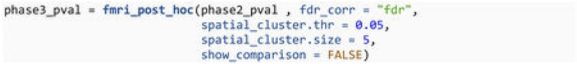
```
phase1_pval = fmri_ROI_phase1(bigim1_mod, mask_label, mask_dict,
                              stimulus_idx = c(1:160)[rep(c(TRUE,FALSE), c(10,10)
)])$all_ROI$pval_t
```

b. Phase2: ROI-Based Tensor-on-Tensor Regression



```
phase2_pval = fmri_ROI_phase2(fmridata = bigim1_mod, label_mask = mask_label, lab
el_dict = mask_dict,
                              stimulus_idx = c(1, 21, 41, 61, 81, 101, 121, 141),
                              stimulus_dur = c(10, 10, 10, 10, 10, 10, 10, 10),
                              rrr_rank = 3, fmri.design_order = 2,
                              fmri.stimulus_TR = 3, method = "t_test")
```

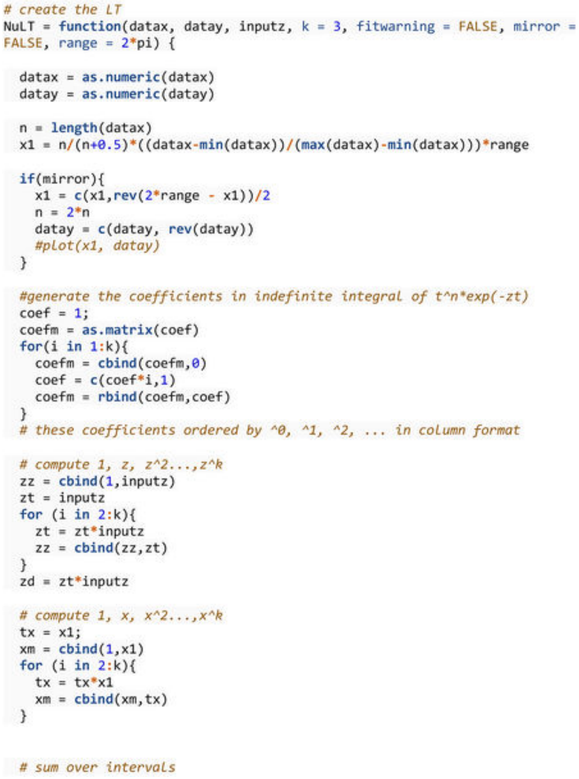
c. Phase3: FDR Correction and Spatial Clustering.



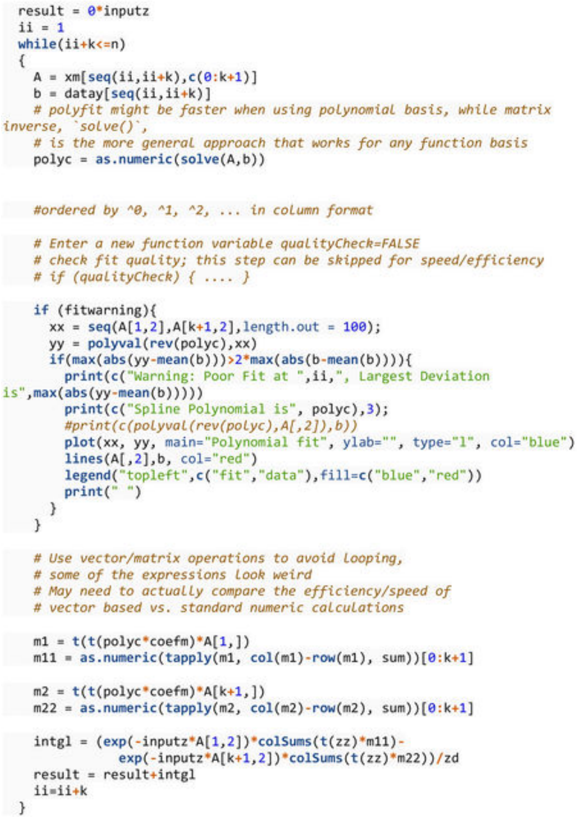
```
phase3_pval = fmri_post_hoc(phase2_pval , fdr_corr = "fdr",
                            spatial_cluster.thr = 0.05,
                            spatial_cluster.size = 5,
                            show_comparison = FALSE)
```

Appendix D.: *R* Implementation of the Laplace Transformation

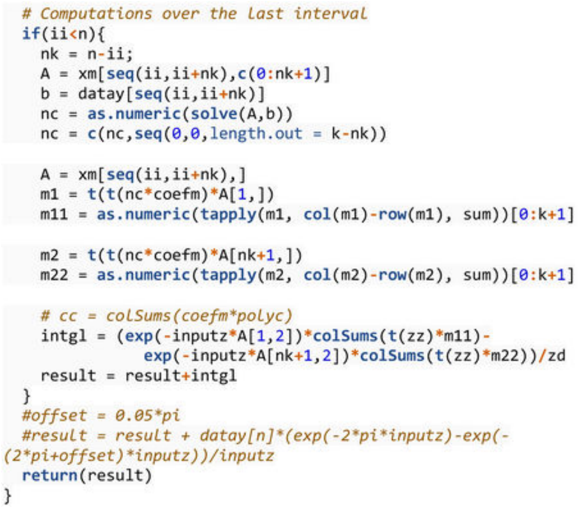
I. Discrete Laplace Transformation



```
# create the LT
NuLT = function(datax, datay, inputz, k = 3, fitwarning = FALSE, mirror =
FALSE, range = 2*pi) {

  datax = as.numeric(datax)
  datay = as.numeric(datay)

  n = length(datax)
  x1 = n/(n+0.5)*((datax-min(datax))/(max(datax)-min(datax)))*range

  if(mirror){
    x1 = c(x1,rev(2*range - x1))/2
    n = 2*n
    datay = c(datay, rev(datay))
    #plot(x1, datay)
  }

  #generate the coefficients in indefinite integral of t^n*exp(-zt)
  coef = 1;
  coefm = as.matrix(coef)
  for(i in 1:k){
    coefm = cbind(coefm,0)
    coef = c(coef*i,1)
    coefm = rbind(coefm,coef)
  }
  # these coefficients ordered by ^0, ^1, ^2, ... in column format

  # compute 1, z, z^2...,z^k
  zz = cbind(1,inputz)
  zt = inputz
  for (i in 2:k){
    zt = zt*inputz
    zz = cbind(zz,zt)
  }
  zd = zt*inputz

  # compute 1, x, x^2...,x^k
  tx = x1;
  xm = cbind(1,x1)
  for (i in 2:k){
    tx = tx*x1
    xm = cbind(xm,tx)
  }


  # sum over intervals
```

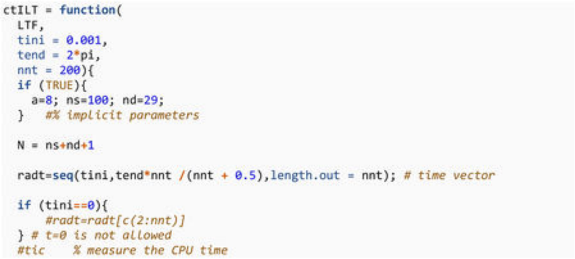


```
  result = 0*inputz
  ii = 1
  while(ii+k<=n)
  {
    A = xm[seq(ii,ii+k),c(0:k+1)]
    b = datay[seq(ii,ii+k)]
    # polyfit might be faster when using polynomial basis, while matrix
inverse, `solve()`,
    # is the more general approach that works for any function basis
    polyc = as.numeric(solve(A,b))


    #ordered by ^0, ^1, ^2, ... in column format

    # Enter a new function variable qualityCheck=FALSE
    # check fit quality; this step can be skipped for speed/efficiency
    # if (qualityCheck) { .... }

    if (fitwarning){
      xx = seq(A[1,2],A[k+1,2],length.out = 100);
      yy = polyval(rev(polyc),xx)
      if(max(abs(yy-mean(b)))>2*max(abs(b-mean(b)))){
        print(c("Warning: Poor Fit at ",ii,", Largest Deviation
is",max(abs(yy-mean(b)))))
        print(c("Spline Polynomial is", polyc),3);
        #print(c(polyval(rev(polyc),A[,2]),b))
        plot(xx, yy, main="Polynomial fit", ylab="", type="l", col="blue")
        lines(A[,2],b, col="red")
        legend("topleft",c("fit","data"),fill=c("blue","red"))
        print(" ")
      }
    }

    # Use vector/matrix operations to avoid looping,
    # some of the expressions look weird
    # May need to actually compare the efficiency/speed of
    # vector based vs. standard numeric calculations

    m1 = t(t(polyc*coefm)*A[1,])
    m11 = as.numeric(tapply(m1, col(m1)-row(m1), sum))[0:k+1]

    m2 = t(t(polyc*coefm)*A[k+1,])
    m22 = as.numeric(tapply(m2, col(m2)-row(m2), sum))[0:k+1]

    intgl = (exp(-inputz*A[1,2])*colSums(t(zz)*m11)-
               exp(-inputz*A[k+1,2])*colSums(t(zz)*m22))/zd
    result = result+intgl
    ii=ii+k
  }
```

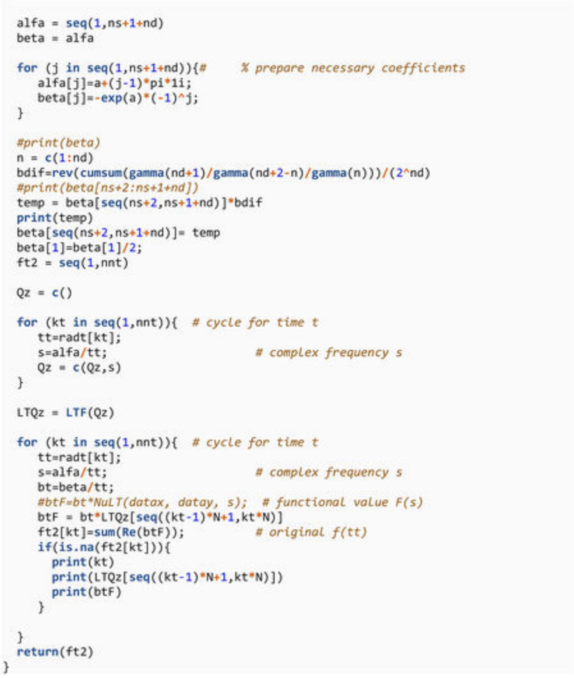


```
  # Computations over the last interval
  if(ii<n){
    nk = n-ii;
    A = xm[seq(ii,ii+nk),c(0:nk+1)]
    b = datay[seq(ii,ii+nk)]
    nc = as.numeric(solve(A,b))
    nc = c(nc,seq(0,0,length.out = k-nk))

    A = xm[seq(ii,ii+nk),]
    m1 = t(t(nc*coefm)*A[1,])
    m11 = as.numeric(tapply(m1, col(m1)-row(m1), sum))[0:k+1]

    m2 = t(t(nc*coefm)*A[nk+1,])
    m22 = as.numeric(tapply(m2, col(m2)-row(m2), sum))[0:k+1]

    # cc = colSums(coefm*polyc)
    intgl = (exp(-inputz*A[1,2])*colSums(t(zz)*m11)-
               exp(-inputz*A[nk+1,2])*colSums(t(zz)*m22))/zd
    result = result+intgl
  }
  #offset = 0.05*pi
  #result = result + datay[n]*(exp(-2*pi*inputz)-exp(-
(2*pi+offset)*inputz))/inputz
  return(result)
}
```

II. Continuous Inverse Laplace Transformation



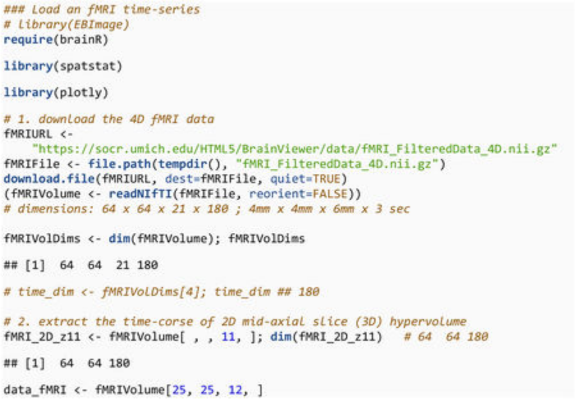
```
ctILT = function(
  LTF,
  tini = 0.001,
  tend = 2*pi,
  nnt = 200){
  if (TRUE){
    a=8; ns=100; nd=29;
  }   #% implicit parameters

  N = ns+nd+1

  radt=seq(tini,tend*nnt /(nnt + 0.5),length.out = nnt); # time vector

  if (tini==0){
      #radt=radt[c(2:nnt)]
  } # t=0 is not allowed
  #tic    % measure the CPU time
```



```
  alfa = seq(1,ns+1+nd)
  beta = alfa

  for (j in seq(1,ns+1+nd)){#     % prepare necessary coefficients
     alfa[j]=a+(j-1)*pi*1i;
     beta[j]=-exp(a)*(-1)^j;
  }

  #print(beta)
  n = c(1:nd)
  bdif=rev(cumsum(gamma(nd+1)/gamma(nd+2-n)/gamma(n)))/(2^nd)
  #print(beta[ns+2:ns+1+nd])
  temp = beta[seq(ns+2,ns+1+nd)]*bdif
  print(temp)
  beta[seq(ns+2,ns+1+nd)]= temp
  beta[1]=beta[1]/2;
  ft2 = seq(1,nnt)

  Qz = c()

  for (kt in seq(1,nnt)){  # cycle for time t
     tt=radt[kt];
     s=alfa/tt;                     # complex frequency s
     Qz = c(Qz,s)
  }

  LTQz = LTF(Qz)

  for (kt in seq(1,nnt)){  # cycle for time t
     tt=radt[kt];
     s=alfa/tt;                     # complex frequency s
     bt=beta/tt;
     #btF=bt*NuLT(datax, datay, s);  # functional value F(s)
     btF = bt*LTQz[seq((kt-1)*N+1,kt*N)]
     ft2[kt]=sum(Re(btF));          # original f(tt)
     if(is.na(ft2[kt])){
       print(kt)
       print(LTQz[seq((kt-1)*N+1,kt*N)])
       print(btF)
     }

  }
  return(ft2)
}
```

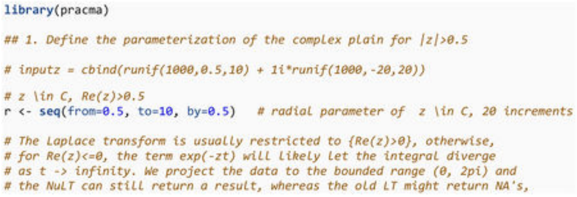
III. fMRI Data Example

This example illustrates testing LT/ITL on real fMRI time-series. First download the data.

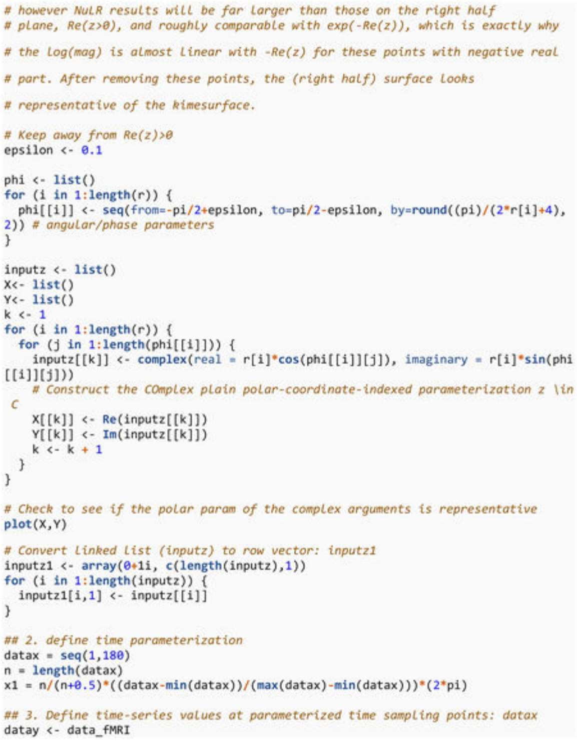


```
### Load an fMRI time-series
# Library(EBImage)
require(brainR)

library(spatstat)

library(plotly)

# 1. download the 4D fMRI data
fMRIURL <-
    "https://socr.umich.edu/HTML5/BrainViewer/data/fMRI_FilteredData_4D.nii.gz"
fMRIFile <- file.path(tempdir(), "fMRI_FilteredData_4D.nii.gz")
download.file(fMRIURL, dest=fMRIFile, quiet=TRUE)
(fMRIVolume <- readNIfTI(fMRIFile, reorient=FALSE))
# dimensions: 64 x 64 x 21 x 180 ; 4mm x 4mm x 6mm x 3 sec

fMRIVolDims <- dim(fMRIVolume); fMRIVolDims

## [1]  64  64  21 180

# time_dim <- fMRIVolDims[4]; time_dim ## 180

# 2. extract the time-corse of 2D mid-axial slice (3D) hypervolume
fMRI_2D_z11 <- fMRIVolume[ , , 11, ]; dim(fMRI_2D_z11)   # 64  64 180

## [1]  64  64 180

data_fMRI <- fMRIVolume[25, 25, 12, ]
```

Define the parametric sampling strategy in the complex plain (ℂ).

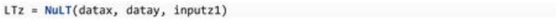


```
library(pracma)

## 1. Define the parameterization of the complex plain for |z|>0.5

# inputz = cbind(runif(1000,0.5,10) + 1i*runif(1000,-20,20))

# z \in C, Re(z)>0.5
r <- seq(from=0.5, to=10, by=0.5)   # radial parameter of  z \in C, 20 increments

# The Laplace transform is usually restricted to {Re(z)>0}, otherwise,
# for Re(z)<=0, the term exp(-zt) will likely let the integral diverge
# as t -> infinity. We project the data to the bounded range (0, 2pi) and
# the NuLT can still return a result, whereas the old LT might return NA's,
```



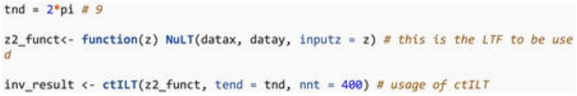
```
# however NuLR results will be far larger than those on the right half
# plane, Re(z>0), and roughly comparable with exp(-Re(z)), which is exactly why

# the log(mag) is almost linear with -Re(z) for these points with negative real

# part. After removing these points, the (right half) surface looks

# representative of the kimesurface.

# Keep away from Re(z)>0
epsilon <- 0.1

phi <- list()
for (i in 1:length(r)) {
  phi[[i]] <- seq(from=-pi/2+epsilon, to=pi/2-epsilon, by=round((pi)/(2*r[i]+4),
2)) # angular/phase parameters
}

inputz <- list()
X<- list()
Y<- list()
k <- 1
for (i in 1:length(r)) {
  for (j in 1:length(phi[[i]])) {
    inputz[[k]] <- complex(real = r[i]*cos(phi[[i]][j]), imaginary = r[i]*sin(phi
[[i]][j]))
    # Construct the COmplex plain polar-coordinate-indexed parameterization z \in
 C
    X[[k]] <- Re(inputz[[k]])
    Y[[k]] <- Im(inputz[[k]])
    k <- k + 1
  }
}

# Check to see if the polar param of the complex arguments is representative
plot(X,Y)

# Convert linked list (inputz) to row vector: inputz1
inputz1 <- array(0+1i, c(length(inputz),1))
for (i in 1:length(inputz)) {
  inputz1[i,1] <- inputz[[i]]
}

## 2. define time parameterization
datax = seq(1,180)
n = length(datax)
x1 = n/(n+0.5)*((datax-min(datax))/(max(datax)-min(datax)))*(2*pi)

## 3. Define time-series values at parameterized time sampling points: datax
datay <- data_fMRI
```

Apply the **NuLT** to fMRI time-series.



```
LTz = NuLT(datax, datay, inputz1)
```

Recover the function of time by applying the **ctILT** to the **NuLT** as analytic complex-valued LTF (function).



```
tnd = 2*pi # 9

z2_funct<- function(z) NuLT(datax, datay, inputz = z) # this is the LTF to be use
d

inv_result <- ctILT(z2_funct, tend = tnd, nnt = 400) # usage of ctILT
```

References:

1. Diggle P, et al., Analysis of longitudinal data. 2002: Oxford University Press.
2. Asar Ö, et al. , Joint modelling of repeated measurement and time-to-event data: an introductory tutorial. International journal of epidemiology, 2015. 44(1): p. 334–344. [PubMed: 25604450]
3. Dinov I, Data Science and Predictive Analytics: Biomedical and Health Applications using R. Computer Science. 2018: Springer International Publishing. 800.
4. Raftery AE, Madigan D, and Volinsky CT, Accounting for model uncertainty in survival analysis improves predictive performance. Bayesian statistics, 1996. 5: p. 323–349.

5. Van der Laan MJ, Laan M, and Robins JM, Unified methods for censored longitudinal data and causality. 2003: Springer Science & Business Media.
6. Dinov I and Velev M, Data Science: Time Complexity, Inferential Uncertainty, and Spacekime Analytics. 1 ed. STEM Series. 2021, Berlin/Boston: De Gruyter. 450 p.
7. Zhou Y, Wong RKW, and He K, Tensor Linear Regression: Degeneracy and Solution. IEEE Access, 2021. 9: p. 7775–7788.
8. Tang M, et al. , Model-Based and Model-Free Techniques for Amyotrophic Lateral Sclerosis Diagnostic Prediction and Patient Clustering. Neuroinformatics, 2018.
9. Marino S, et al. , Compressive Big Data Analytics: An Ensemble Meta-Algorithm for High-dimensional Multisource Datasets. PLoS, 2020. 15(8): p. e0228520.
10. Nesselroade JR and Baltes PB, Longitudinal research in the study of behavior and development. 1979: Academic Press.
11. Grimm KJ, Davoudzadeh P, and Ram N IV. Developments in the analysis of longitudinal data. Monographs of the Society for Research in Child Development, 2017. 82(2): p. 46–66.
12. Fitzmaurice GM and Ravichandran C, A primer in longitudinal data analysis. Circulation, 2008. 118(19): p. 2005–2010. [PubMed: 18981315]
13. Telzer EH, et al. , Methodological considerations for developmental longitudinal fMRI research. Developmental cognitive neuroscience, 2018. 33: p. 149–160. [PubMed: 29456104]
14. Taris T, Longitudinal data analysis. 2000: Sage.
15. Wei WW, Time series analysis, in The Oxford Handbook of Quantitative Methods in Psychology: Vol. 2. 2006.
16. Verbeke G, et al. , The analysis of multivariate longitudinal data: a review. Statistical methods in medical research, 2014. 23(1): p. 42–59. [PubMed: 22523185]
17. Perea J, Topological Time Series Analysis. Notices of the American Mathematical Society, 2019(5): p. 686–694.
18. Boissonnat J-D, Chazal F, and Yvinec M, Geometric and topological inference. Vol. 57. 2018: Cambridge University Press.
19. Kennedy SM, Roth JD, and Scrofani JW. A novel method for topological embedding of time-series data. in 2018 26th European Signal Processing Conference (EUSIPCO). 2018. IEEE.
20. Bao W, Yue J, and Rao Y, A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PloS one, 2017. 12(7): p. e0180944. [PubMed: 28708865]
21. Friston KJ, et al. , Analysis of fMRI time-series revisited. Neuroimage, 1995. 2(1): p. 45–53. [PubMed: 9343589]
22. Aron AR, Gluck MA, and Poldrack RA, Long-term test–retest reliability of functional MRI in a classification learning task. Neuroimage, 2006. 29(3): p. 1000–1006. [PubMed: 16139527]
23. Skup M, Longitudinal fMRI analysis: A review of methods. Statistics and its interface, 2010. 3(2): p. 235–252. [PubMed: 21691445]
24. Kumar N, et al., IEEE 12th International Conference on Computer Vision. 2009, IEEE.
25. Lock EF, Tensor-on-tensor regression. Journal of computational and graphical statistics : a joint publication of American Statistical Association, Institute of Mathematical Statistics, Interface Foundation of North America, 2018. 27(3): p. 638–647.
26. De Martino F, et al. , Predicting EEG single trial responses with simultaneous fMRI and relevance vector machine regression. Neuroimage, 2011. 56(2): p. 826–36. [PubMed: 20691274]
27. Acar E, et al. Tensor-based fusion of EEG and FMRI to understand neurological changes in schizophrenia. in 2017 IEEE International Symposium on Circuits and Systems (ISCAS). 2017. IEEE.
28. Ferdowsi S, Abolghasemi V, and Sanei S, A new informed tensor factorization approach to EEG–fMRI fusion. Journal of neuroscience methods, 2015. 254: p. 27–35. [PubMed: 26231620]
29. Belding WR, Bases for the Positive Cone of a Partially Ordered Module. Transactions of the American Mathematical Society, 1978: p. 305–313.
30. Robinson DJ, An introduction to abstract algebra. 2008: de Gruyter.
31. Anderson E, The Problem of Time. 2017: Springer.

32. Spiegel MR, Laplace transforms. 1965: McGraw-Hill New York.

33. Adler RJ, The geometry of random fields. 2010: SIAM.

34. Worsley K, Local maxima and the expected Euler characteristic of excursion sets of X2, F and T fields. Advances in Applied Probability, 1994. 26: p. 13–42.

35. Valsa J and Brančik L, Approximate formulae for numerical inversion of Laplace transforms. International Journal of Numerical Modelling: Electronic Networks, Devices and Fields, 1998. 11(3): p. 153–166.

36. Berenstein CA and Yger A, Residue calculus and effective Nullstellensatz. American Journal of Mathematics, 1999. 121(4): p. 723–796.

37. Abramowitz M, Stegun IA, and Romer RH, Handbook of mathematical functions with formulas, graphs, and mathematical tables. 1988, American Association of Physics Teachers.

38. Peltier SJ, et al. Support vector machine classification of complex fMRI data. in 2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society. 2009.

39. Peltier S, et al. Multivariate Classification of Complex and Multi-echo fMRI Data. in 2013 International Workshop on Pattern Recognition in Neuroimaging. 2013.

40. Tabelow K and Polzehl J, Statistical parametric maps for functional MRI experiments in R: The package fmri. 2010: WIAS.

41. Eloyan A, et al. , Analytic Programming with fMRI Data: A Quick-Start Guide for Statisticians Using R. PLOS ONE, 2014. 9(2): p. e89470. [PubMed: 24586801]

42. Shattuck DW, et al. , Construction of a 3D probabilistic atlas of human cortical structures. NeuroImage, 2008. 39(3): p. 1064–1080. [PubMed: 18037310]

43. Mandal PK, Mahajan R, and Dinov ID, Structural Brain Atlases: Design, Rationale, and Applications in Normal and Pathological Cohorts. Journal of Alzheimer's Disease, 2012.

44. Tang Y, et al. , The construction of a Chinese MRI brain atlas: A morphometric comparison study between Chinese and Caucasian cohorts. Neuroimage, 2010. 51(1): p. 33–41. [PubMed: 20152910]

45. Welvaert M and Rosseel Y, On the definition of signal-to-noise ratio and contrast-to-noise ratio for fMRI data. PloS one, 2013. 8(11): p. e77089. [PubMed: 24223118]

46. Geissler A, et al. , Contrast-to-noise ratio (CNR) as a quality parameter in fMRI. Journal of Magnetic Resonance Imaging: An Official Journal of the International Society for Magnetic Resonance in Medicine, 2007. 25(6): p. 1263–1270.

47. Fay MP and Proschan MA, Wilcoxon-Mann-Whitney or t-test? On assumptions for hypothesis tests and multiple interpretations of decision rules. Statistics surveys, 2010. 4: p. 1. [PubMed: 20414472]

48. Benjamini Y and Hochberg Y, Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. Journal of the Royal Statistical Society. Series B (Methodological), 1995. 57(1): p. 289–300.

49. Lohmann G, et al. , LISA improves statistical analysis for fMRI. Nature Communications, 2018. 9(1): p. 4014.

50. Wu CO and Tian X, Nonparametric models for longitudinal data: With implementation in R. 2018: CRC Press.

51. Van Montfort K, Oud JH, and Voelkle MC, Continuous time modeling in the behavioral and related sciences. 2018: Springer.

52. Molenberghs G and Verbeke G, Models for Discrete Longitudinal Data. 2006: Springer New York.

53. Banus J, et al. , Biophysics-based statistical learning: Application to heart and brain interactions. Medical Image Analysis, 2021. 72: p. 102089. [PubMed: 34020082]

54. Abbas IA and Marin M, Analytical Solutions of a Two-Dimensional Generalized Thermoelastic Diffusions Problem Due to Laser Pulse. Iranian Journal of Science and Technology, Transactions of Mechanical Engineering, 2018. 42(1): p. 57–71.

55. Marin M, et al. , A domain of influence in the Moore–Gibson–Thompson theory of dipolar bodies. Journal of Taibah University for Science, 2020. 14(1): p. 653–660.

56. Kolda TG and Bader BW, Tensor decompositions and applications. SIAM review, 2009. 51(3): p. 455–500.

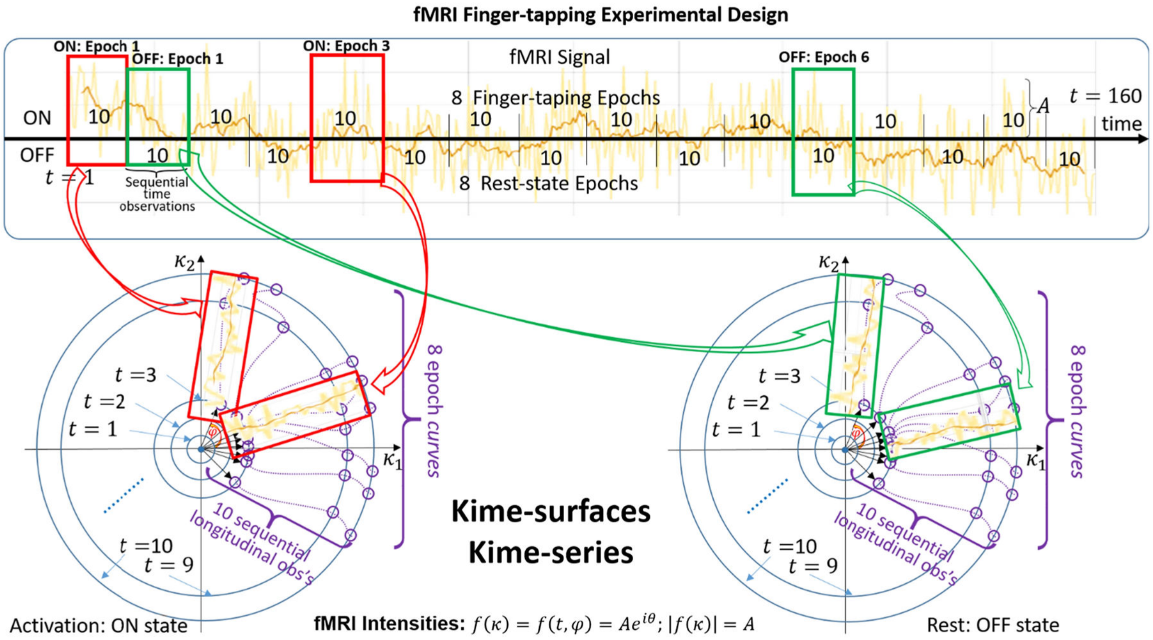


Figure 1.

Schematic diagram of an empirical mapping between an fMRI time-series and its corresponding kimesurface.

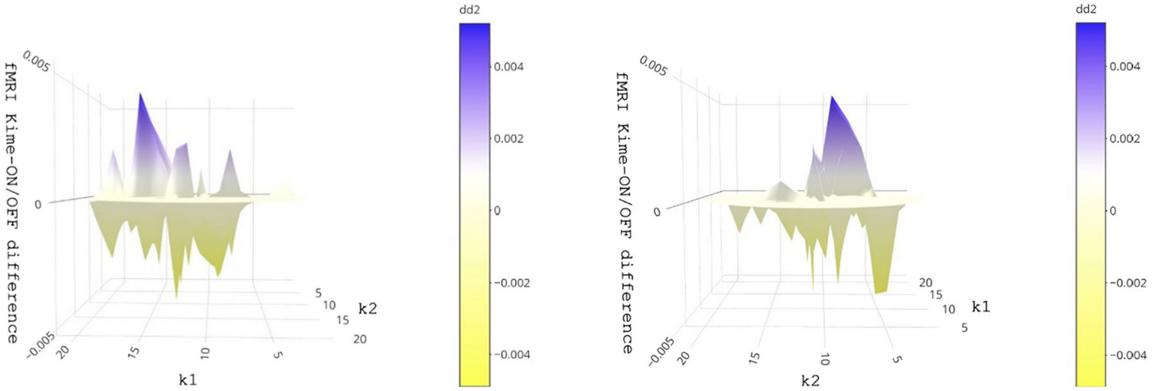


Figure 2.

A pair of 3D surface renditions of the ON-OFF fMRI kimesurface differences at a fixed spatial location. Blue and yellow colors indicate positive and negative kimesurface differences, respectively.

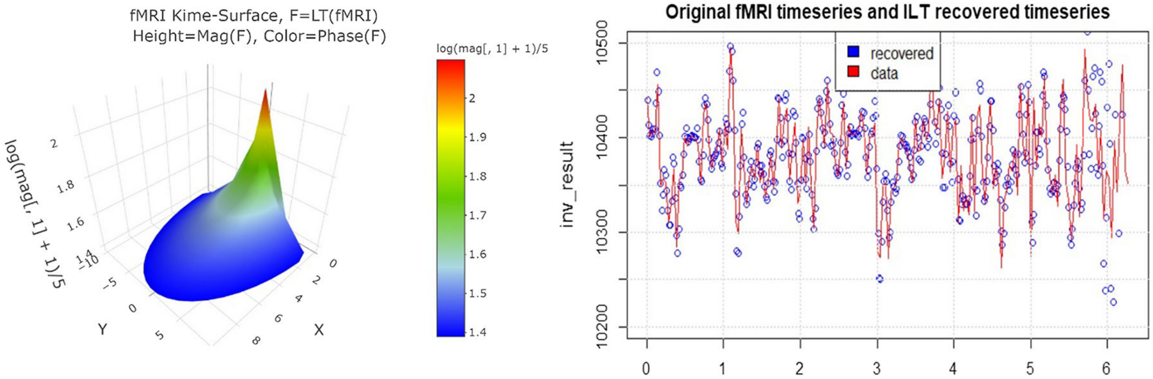


Figure 3.

The duality between a real fMRI time-series (red curve on the right panel), its corresponding kimesurface (left panel), and the reconstructed fMRI time-source using the ILT on the kimesurface (blue scatterplot on the right panel).

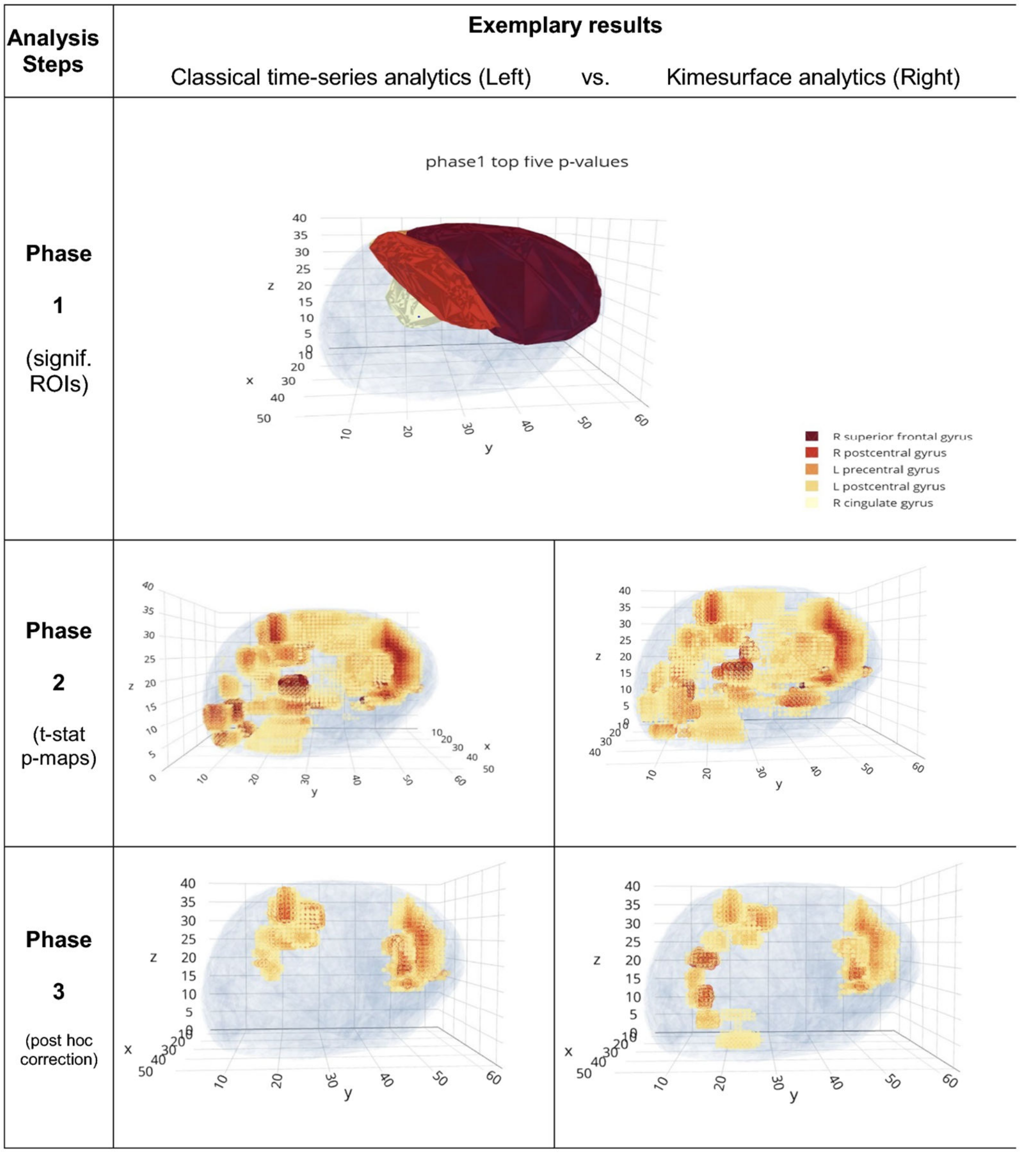


Figure 4.

Results of the three-phase statistical analysis protocol on native fMRI time-series data (*left*) and the corresponding fMRI kimesurface representations (*right*). The 3D renderings illustrate different point-of-view orientations. *Note*: The Phase 1 results for the classical (left) and proposed space-kime analytics are identical, as the ROI-analytics are done on the raw space-time data.