Least Squares-Optimal Transport (LS-OT) Regression for varying time delays

B. Tsakam-Sotché, brice.tsakam@gmail.com Horgen, revised on August 17th 2024

Abstract

While analysing time series data, conventional linear regression methods leave us the choice with one of two assumptions: either the dependent and independent variables are synchronous or fixed time lags are assumed.
However the time lag might be varying over time, or noisy. The time lag becomes an additional source of error beside the usual (measurement) noise. Ignoring the variable time lag amounts to a specification error. We use optimal transport cost as a way to account for the time lag uncertainty and make ordinary least squares (OLS) robust to this type of error or noise. Using the chain rule, this enhancement to the conventional OLS regression model of noise easily generalises to multivariate regression.

Introduction

We investigate the simple regression for non-synchronous time series. This type of data is common in practice in a variety of contexts. We propose a method that is relevant when all the usual assumptions for OLS are fulfilled but there is still an incorrect model specification in the sense that the time lag variation is significant in comparison to the variation of the dependent variable. This applies to situations where the dependent and independent variables are connected with a transmission link that is not instantaneous and might even exhibit shocks in the transmission delay.

Current methods for this situation include OLS with multiple time lags, but with time lag averaging to zero, the best regressor is one that assumes no time lag at all. Hence we compare the proposed method with simple OLS regression with no time lag assumption as a baseline.

OT distance, and optimal transportation plan are robustly calculated even for a very large number of support points [9]. While fast solvers for the optimal transport (OT) cost are available, the corresponding gradient is still challenging to compute. One convenient approach is to use some flavour of automatic differentiation [7, 8]. In practice this approach proves numerically unstable as the number of support points in the distribution increases beyond a few dozens.

As an alternative, we use the transportation plan to derive an OT gradient and the corresponding chain rule. Starting with simple regression as an example and then generalising to functions only required to be continuously differentiable in all their parameters.

First, we review OLS regression, Optimal Transport and the proposed Least-Square Optimal-Transport (LS-OT) regression in Section 1. Section 2 presents the experimentation methodology with synthetic data and we compare OLS and LS-OT regression results in Section 3. We conclude with a discussion of the results, limitations and further research.

Section 1

1.1 OLS regression

We first recall OLS basic assumptions[2| given observed data point (Y,X):

- The dependent variable is modelled as linear function of the independent variables: y = slope * X
- The observations (Xi,Yi) are independent and identically distributed (iid)
- The non-dependent variables are linearly independent
- residuals eta = Y-y are normal, uncorrelated with bounded variance

The OLS [2] method aims at inferring the dependent variable y_t from the contemporary independent variable xt and time lags thereof x_{t-1}, x_{t-2}, etc. The linear dependency factor slope, which is identified through an optimization process that minimises OLS_{cost} defined as follows:

$$OLS_{cost} = \Sigma_t (Y_t - slope \times X_t)^2$$

Regression fit can be evaluated using the coefficient of determination r-squared which is related to the ratio of unexplained over total variance:

$$R2 = 1 - \Sigma_t (Y_t - y_t)^2 / \Sigma_t (Y_t - E(Y_t))^2$$

R2 = 1 in case of a perfect fit and R2 = 0 when none of the variance is explained by the model.

1.2 Optimal Transport

Let's consider the Optimal Transport cost [1]:

$OT_{cost} = \Sigma\, Cost(X, T(X))$ for which the optimal plan for which $T^*(M)=V$, *M*

M and *V* are measures defined on the support set of points {X}. $T^*(M)$ is the optimal transportation plan that moves the mass of *M* into *V* at a minimal cost given the Cost matrix C.

In comparison, the OLS uses the Euclidean distance as cost and sets T to be a linear function parameterized with Slope: T(X) = Slope * X.
We observe that this is identical to the Wasserstein distance when the transport cost is defined by the Euclidean distance except that no error is allowed in the time axis which can be seen as an infinite cost.

Section 2

2.1 Least-Squares Optimal-Transport regression

Now in the OT distance minimization framework, we have the flexibility to allow a time lag and handle data that is not perfectly synchronous. We state the regression problem as an optimization problem combining the OLS and OT costs, with a trade-off parameter lambda:
$argmin(slope) : OLS_{cost}(slope) + lambda \times OT_{cost}(slope)$

Where the OT cost calculation is parameterized with an isotropic cost matrix. As will be shown in Section 1.4, the OT cost matrix can be parameterized to include the OLS cost and the trade-off between OLS and OT costs. This simplifies the formulation to:
$argmin(slope) : cst \times OT_{cost}(slope)$

2.1 Gradient decomposition and chain rule

We start with an univariate regression. Given an input set of points (X,Y) and initial estimated parameters for the slope and intercept also represented by a set of points (x,y). We want to find the regression parameters that minimises the OT distance between the supports (X,Y) and (x,y). We obtain the transportation plan from the backAndForth solver [source back and forth]. It is composed of 2 dimensions (e_1, e_2). We use a chain rule to back-propagate the OT distance gradient through both dimensions. Then we apply the chain rule for each dimension separately.

The e_2 direction is the familiar way of understanding the gradient of a 1 argument value function, that is along the y-axis. Considering a transportation plan in the e_2 direction (ie using a non-isotropic transport cost where displacements along e_1 are prohibitively elevated) amounts to a conventional OLS regression.

Then, the e_1 direction is all about displacements along the x-axis. This requires a slightly different handling as will be shown starting with the following example.

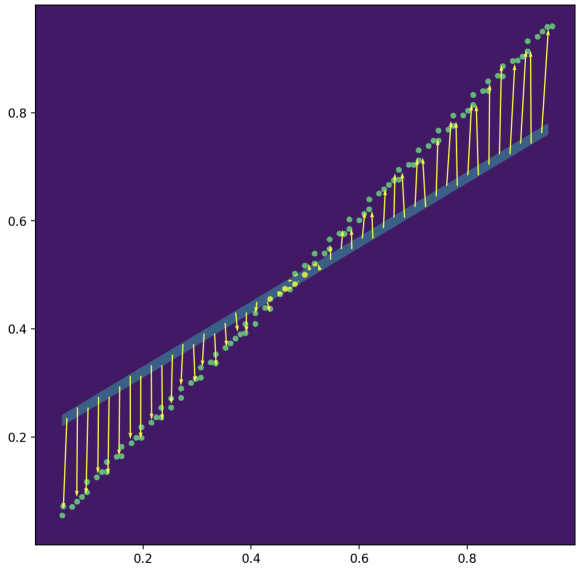

Figure 1: OT plan from the estimated solution to the data to fit

2.1.1 Linear regression with parameters slope and intercept

We start with 2 sets of points: the starting support points defined by the pairs (x,y) generated as per the parameters $slope_{est}$, b_{est}. The destination support points defined by the pairs (X,Y) which is the given data and for which we search the true regression parameters $slope,\ b$. The true relationship is defined as:

$Y = slope_{true} \times X + b_{true}$

Current estimates of the parameters give us

$y = slope_{est} \times x + b_{est}$

The transportation plan is defined in 2 dimensions, along the x-axis (e_1) and along the y-axis (e_2). When looking at displacements along the x-axis, we index the y-axis according to Y_j. Thus defining the x-axis coordinate as the first member of the pair $(., y_j)_{est}$ as per pair's relationship definition. And similarly when looking at displacements along the y-axis. From here on we drop the $_{-true}$ subscript for simplicity.

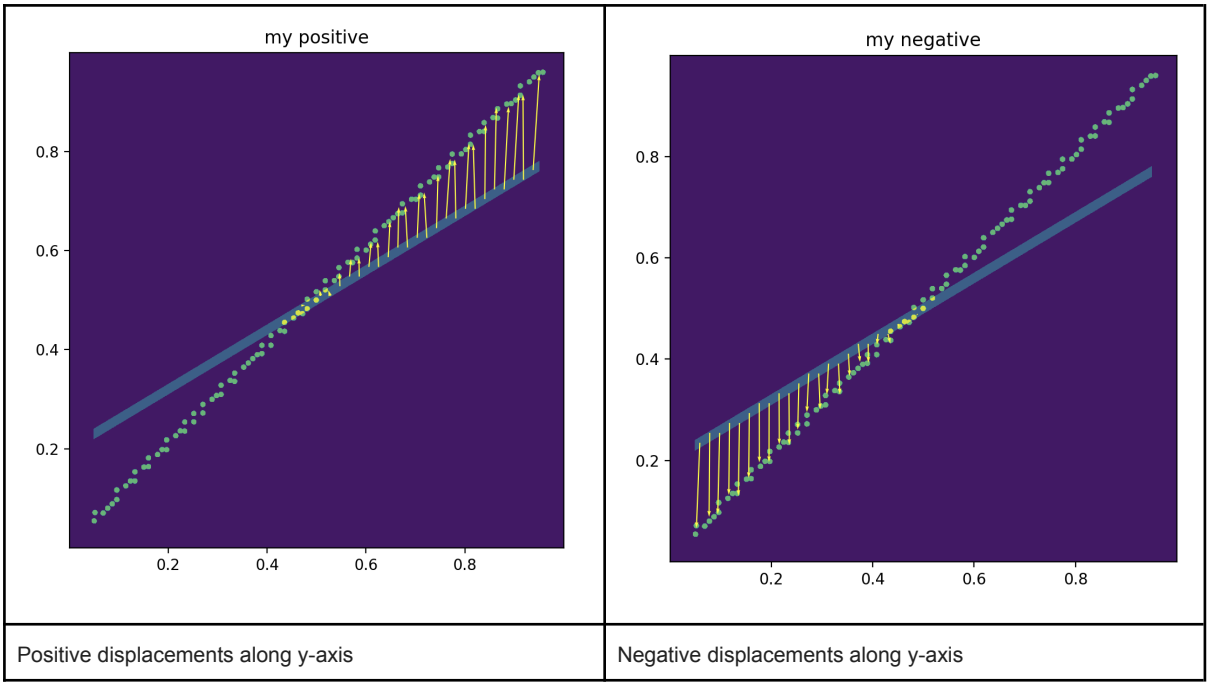

Figure 2: OT transportation plan positive and negative displacements

2.1.1.1 Parameter inference with regression

Visual inspection of the displacements suggests that their amplitude can be regressed to find the gradient to the slope and to the intercept. We observe that for the displacements along y-axis (direction e_2), the terms Δslope and Δb can be obtained by linear regression by solving:

$$\Delta y = \Delta slope_{est} \times x + \Delta b_{est}$$

Where Δy_j at each x_j is given by the transportation plan.

Along the x-axis (direction e_1) we have:

$$\Delta x = \Delta inverseSlope \times y + \Delta inverseB$$

Here, Δx and y are given by the transportation plan and the previous estimate or Y, respectively. Again, using a conventional regression we obtain the parameters $\Delta inverseSlope,\ \Delta inverseB$

Now, using the inverse slope (obtained at previous step) $1/slope_{est}$, the newly calculated optima slope is then:

$$inverseSlope_{new\ est} = (1/slope_{est} + \Delta inverseSlope)$$

$$slope_{new\ est} = 1/inverseSlope_{new\ est}$$

$$\Delta slope = slope_{new\ est} - slope_{est} = 1/(1/slope_{est} + \Delta inverseSlope) - slope_{est}$$

And the newly calculated optimal b is then
$\Delta b = -(1/slope_{est} + \Delta inverseSlope) * \Delta inverseB$

2.1.1.2 Parameter inference details

Slope parameter

Given the transportation plan in the **direction** e_2 is ΔY, from y to Y
$Y - y = slope \times X + b - (slope_{est} \times x + b_{est})$

$Y - y = (slope - slope_{est}) \times X + (b - b_{est})$

If we ignore the b parameter this amounts to $\Delta slope = \Delta Y / X$ that we estimate a
$argmin(\Delta slope) : \boldsymbol{\Sigma}(\Delta slope \times X - \Delta Y)^2$

Given the transportation plan in the **direction** e_1, is ΔX, from x to X. We now have
$Y = slope \times X + b = slope_{est} \times x + b_{est}$

With the direction e_1: y is unchanged. This part of the transportation plan describes only changes along the x-axis.
$slope = (slope_{est} \times x + b_{est} - b)/X$

$slope - slope_{est} = (slope_{est} \times x + b_{est} - b)/X - slope_{est}$

$\Delta slope = (y - b)/X - slope_{est}$

Which we estimate as:
$argmin(\Delta slope) : \boldsymbol{\Sigma}(\Delta slope - (y - b)/X + slope_{est})^2$

Intercept parameter

For simplicity, we assume that the slope is constant or frozen
For the direction e_2:
$Y - y = slope \times X + b - (slope_{est} \times x + b_{est}) = \Delta b$

Ignoring the slope parameter, we need to solve:
$argmin(\Delta b) : \boldsymbol{\Sigma}(\Delta b - (Y - y))^2$

For the direction e_1:
Given the x-axis displacements $\Delta X = X - x$ and no change on the y-axis
$Y = slope \times X + b = slope_{est} \times x + b_{est} = y$

$$b - b_{est} = slope_{est} \times x - slope \times X$$

Here we use our current estimate of slope and the previous estimate
$$b - b_{est} = slope_{est} \times x - slope_{est\ new} \times X$$

$$\Delta b = slope_{est} \times x - slope_{est\ new} \times X$$

Which is estimated as
$$argmin(\Delta b) : \boldsymbol{\Sigma}(\Delta b - (slope_{est} \times x_j - slope_{est\ new} \times X_j))^2$$

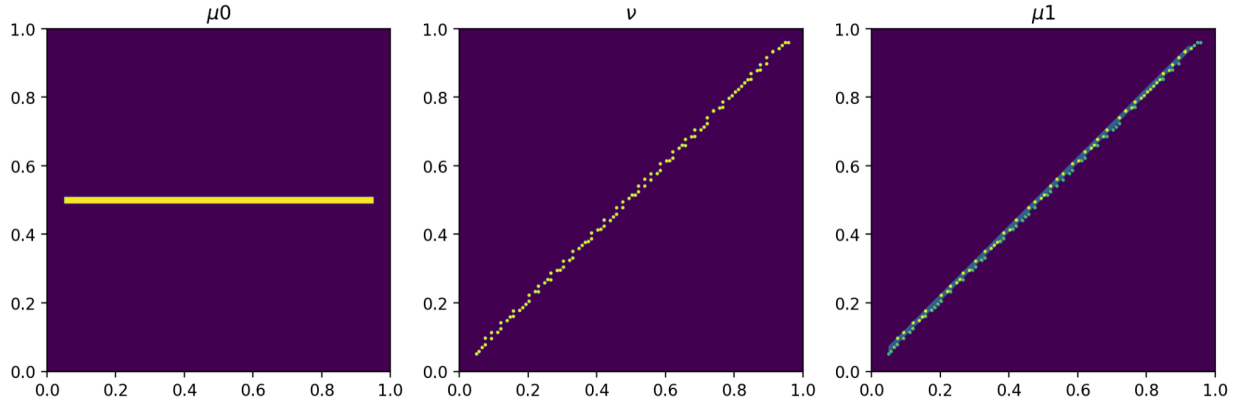

Figure 3: left: initial y. middle: Y. right: solution for y overlaid on Y.

2.1.2 Continuously derivable function with one argument

We define a 1 time continuously derivable function parameterized by θ and accepting one argument x:
$y = f(\theta, x)$ and its derivative wrt θ

$$\delta y/\delta\theta = \delta f(\theta, x)/\delta\theta$$

We start with input observations (X,Y) which we assume are governed by the relationship
$Y = f(\vartheta, X)$ with some unknown ϑ

Given the transportation plan in the direction e_2 is $\Delta Y = Y - y$
$\Delta Y = \delta f(\theta, x)/\delta\theta \times \Delta\theta$

Hence we need to solve:
$$argmin(\Delta\theta) : \boldsymbol{\Sigma}(\Delta Y - \delta f(\theta, x)/\delta\theta \times \Delta\theta)^2$$

Given the transportation plan in the direction e_1, is ΔX, from x to X
$Y = f(\vartheta, X) = f(\theta, x) = y$

Using the Taylor rule for $f(\vartheta, X) = f(\theta, X) + \delta f(\theta, X)/\delta\theta \times \Delta\theta$
We rewrite
$Y = f(\theta, X) + \delta f(\theta, X)/\delta\theta \times \Delta\theta = f(\theta, x)$

Here we need to solve:
$argmin(\Delta\theta) : \boldsymbol{\Sigma}_i (f(\theta, X_i) + \delta f(\theta, x_i)/\delta\theta \times \Delta\theta - y_i)^2$, where $f(\theta, x_i) = y_i$

1.4.2.1 Multiple argument functions

The generalisation of this chain rule for OT distance transport plan back-propagation to continuously derivable functions with several arguments (or in multiple dimensions) is straightforward if all argument variables share the same innovation in time delays.

Turning to a function of multiple arguments with non-synchronous time delays:
$y = f(\theta, x1, x2, ..., x_N)$ where each argument x_j has its own time delay innovation variable ε_j. For this case we use the Sinkhorn barycenter approach [4] where we calculate the gradient in turn for each of the input argument and apply the resulting gradient step until we converge to the optimal parameter, as close as possible to ϑ.

Section 3

3.1 Synthetic data generator

In the case of OLS the input data is assumed to be synchronous. The time lag is assumed to be fixed. We now consider a data generation process with explicit random time lag:
y_t = beta * X(t+eta_t) + eta

Where X_t is a vector: [x_t, 1], beta is a constant vector: [slope, intercept]
Eta is normal with variance s_e. The time lag noise eta_t is normal with variance s_e_t

Equipped with this data generator, we proceed to estimate beta with different methods and compare results.

3.2 Implementation details

For the first experiment, use python jax [3] automatic differentiation software for the OT cost gradients. The chain rule allows to backpropagate the gradient to regression model parameters, then gradient descent converges to the desired minima. We compensate for the bias in the OT distance calculation as in [4]. We run 100 simulations and report estimated slope parameters using the OLS and the Transport regression.

(The complete jupyter notebook for data generation and regression experiments presented here is available at: [https://colab.research.google.com/drive/1MH_vGrtf-cuPKtxBQbokLsDRRrXr5ero])

In a second experiment, we use the back and forth OT solver together with the OT gradient chain rule presented in section 1.4 [6]. This solver calculates sparse plans and scales to thousands of points in the source and destination support point sets. We run 50 simulations and report the slope parameters using OLS and this second method for the gradient.
(The complete jupyter notebook for data generation and regression experiments presented here is available at: [https://colab.research.google.com/drive/1GhiKSw8g7rqstvEfQ0eOjIMMQG4slxjM])

In both experiments we allow only positive time lags (i.e. positive time lag noise eta_t).

Section 4

4.1 Results

As shown in Figure 1, automatic differentiation method, brings significant improvement on OLS alone for the regression task with a time varying lag (where the lag standard deviation is similar to several regular time steps between samples). We observe that the improvement for most samples except in the few cases where the slope estimate overshoots the true value. In most of the 100 experiments, both estimators undershoot the true value but LS-OT is robustly more accurate.

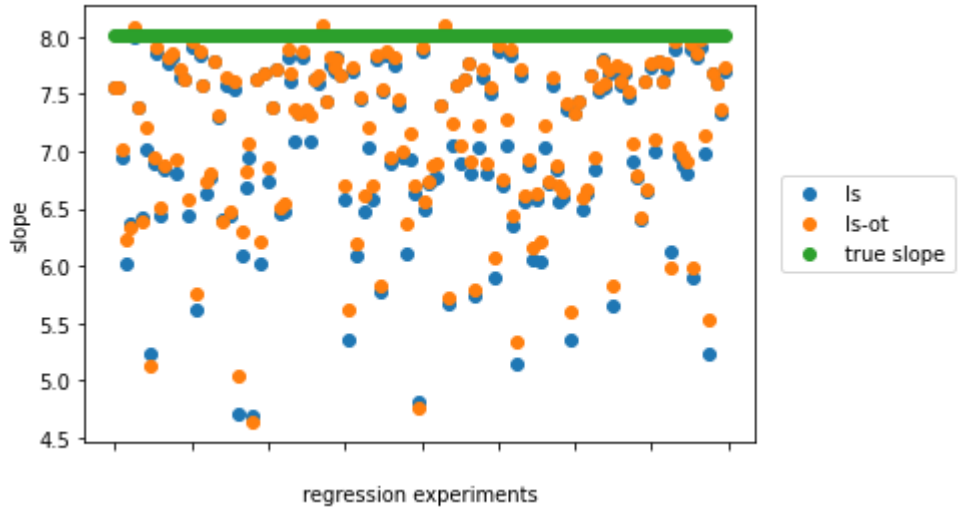

Figure 4: ls shows OLS estimated Beta, LS-OT fit using automatic differentiation, and the true slope (generator parameter)

The results of the second method are shown in Figure 2, using OT transportation plan based gradient. Here, the second method brings significant improvement on

OLS alone for the regression task with a time varying lag (where the lag standard deviation is similar to one regular time step between samples).

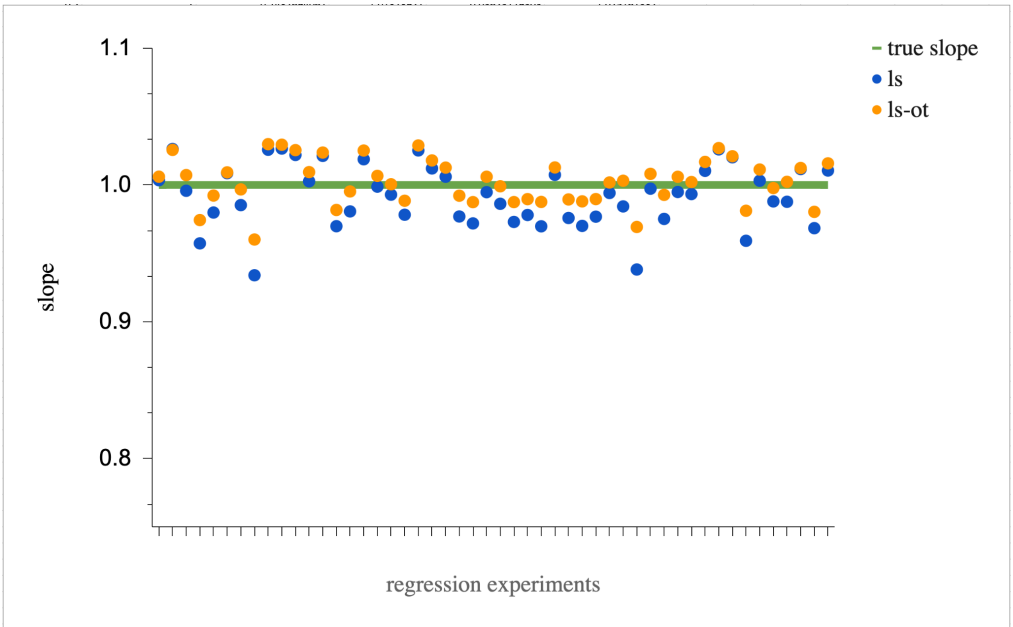

Figure 5: OLS estimated Beta, LS-OT fit using OT transportation plan and chain rule as gradient, and the true slope (data generator parameter)

We observe that the LS-OT method is a favourable option when non-synchronous data is at hand and the preferred model when expert knowledge on the time lag variance is available.

4.2 Discussion

We presented the LS-OT solver for regressions where the dependent variable gets information from the independent variable with a variable delay, modelized by a gaussian variable. This is relevant in many practical settings where multivariate analysis includes several time series of connected but disjoint phenomena in biology, mechanics, finance and economics, etc. We presented a solver that scales to thousands of points in the source and destination supports, while the automatic gradient based method exhibits numerical stability issues already with 100 points in the support. This method can easily be extended to partial least square, autoregressive models that exhibit variable time lags and vector variants thereof. Finally, we observe that in addition to informing regression parameters, the transportation plan gives cues about timing relationships among the variables, which is an avenue for further research.

4.3 Further research

The automatic differentiation method is severely limited by the OT accuracy and numerical stability. It can be applied for very short times eries with 10-20 samples because of the limited accuracy of the gradient descent OT solver.

The second method with OT transportation plan based gradient is much more scalable with thousands of samples and robust numerical stability and accuracy as shown in the second experiment. It is a practical implementation suitable for real life regression problems. It combines OLS parameter inference and dynamic time warping [10].
A graphical observation of the transport plan with synthetic data reveals that the dependent variable is always delayed (on the x-axis) with our synthetic data. Statistical mean and standard deviation of the components of the optimal transportation plan gives an estimate of the delay generator parameters. This indicates that the LS-OT method gives cues about time based precedence or causality among the variables. The ability of LS-OT to account for timing suggests novel analysis of lead-lag relationships. We leave these aspects for future research.

In addition, we may apply the singular value decomposition (SVD) on the transportation plan to get the main (largest) eigenvalues and corresponding eigenvectors as a rough characterisation of the time lag. A single mode (of fast decaying eigenvalues) SVD would suggest that it is better to revert to OLS with the appropriate fixed time lag while multiple significant eigenvalues means that the time lag is arguably not fixed and diverse.

References

[1] Wikipedia article for Optimal Transport. Retrieved in February 2023 https://en.wikipedia.org/wiki/Transportation_theory_(mathematics)#Abstract_formulation_of_the_problem

[2] Wikipedia article for Ordinary Least Squares. Retrieved in February 2023 https://en.wikipedia.org/wiki/Ordinary_least_squares

[3] Github repo for jax (February 2023) https://github.com/google/jax

[4] Hicham Janati, Marco Cuturi, Alexandre Gramfort, “Debiased Sinkhorn barycenters”, retrieved Feb 2023 https://arxiv.org/abs/2006.02575

[5] Matt Jacobs and Flavien Léger, A fast approach to optimal transport: the back-and-forth method, Numer. Math. 146 (2020), no. 3, 513–544, doi:10.1007/s00211-020-01154-8.

[6] Implementation of Back-and-forth method in optimal transport https://github.com/Math-Jacobs/bfm

[7] JAX https://jax.readthedocs.io/en/latest/notebooks/quickstart.html

[8] Autograd automatic differentiation for numpy https://github.com/HIPS/autograd

[9] Marco Cuturi, “Sinkhorn Distances: Lightspeed Computation of Optimal Transport”, NIPS 2013

[10] H. Sakoe and S. Chiba, “Dynamic programming algorithm optimization for spoken word recognition”. ICASSP 1978