Abstract—Using least-squares with an $l_1$-norm penalty is well-known to encourage sparse solutions. In this article, we propose an algorithm that performs online least-squares estimation of a time varying system with a $l_1$-norm penalty on the variations of the state estimate, leading to state estimates that exhibit few “jumps” over time. The algorithm analytically computes a path to update the state estimate as a new observation becomes available. The algorithm performs computationally efficient and numerically robust state estimation for time varying systems in which the dynamics are slow compared to the sampling rate.

We use the algorithm for arterial traffic estimation with streaming probe vehicle data provided by the Mobile Millennium system and show a significant improvement in the estimation capabilities compared to a baseline model of traffic estimation. The estimation framework filters out the inherent noise of traffic dynamics and improves the interpretability and accuracy of the results. Results from an implementation in San Francisco on a network of more than 800 links using a fleet of 500 taxis sending their location every minute illustrate the possibility to use the algorithm to solve important practical estimation problems.

I. PROBLEM STATEMENT AND RELATED WORK

$L_1$-norm in least-squares regression has attracted a lot of interest in the statistics [1], [2], signal processing [3], [4] and machine learning [5], [6] communities, in particular for estimation problems. Indeed, adding a $l_1$-penalty on the parameter vector leads to sparse solutions, which is a desirable property in order to achieve model selection [7], data compression [4], or for obtaining interpretable results. In this article, we present a way to use $l_1$-norm regularization to perform estimation of a time varying system. We assume that we receive sequential observations of the state of the system. As we receive a new observation, we update the estimate of the state online and we would like the variations in the estimates to be sparse.

We are sequentially given a set of training examples or observations $(y_i,a_i) \in \mathbb{R} \times \mathbb{R}^m$, $i = 1...n$. We wish to fit a linear model to estimate the response $y_i$ as a function of the state vector $x \in \mathbb{R}^m$, $y_i = a_i^T x + v_i$, where $v_i$ represents the noise in the observation.

Least square optimization with a penalty on the $l_1$-norm of the parameter is known as the Lasso algorithm [1] and the resulting optimization problem is given by

$$x = \arg\min_{x \in \mathbb{R}^m} \frac{1}{2} \sum_{i=1}^{n} (a_i^T x - y_i)^2 + \mu_n ||x||_1$$

where $\mu_n$ is a regularization parameter. The solution of (1) is typically sparse, i.e. the solution has few entries that are non-zero, and therefore identifies which dimensions in $a_i$ are useful to predict $y_i$. In model selection for example, the different elements of $a_i$ represent different features and the $l_1$ regularization selects the most relevant features to estimate $y_i$. There is no analytic formula for the optimal solution to the $l_1$-regularized least square. It is convex but not differentiable and requires specific algorithms to be solved efficiently at a large scale. It can be formulated as a convex quadratic problem (QP) with linear equality constraints and solved using standard interior-point methods which can handle medium-size problems [8]. A specialized interior-point method for large-scale problems was introduced in [9]. Other methods to solve (1) include iterative thresholding algorithms [10], [11], [12], feature-sign search [13], bound optimization methods [14] and gradient projection algorithms [15]. Homotopy methods have also been applied to the Lasso to compute the full regularization path when the regularization parameter $\mu_n$ varies [16], [17], [18]. They are particularly efficient when the solution is very sparse [19]. When the training examples $(y_i,a_i)_{i=1...n}$ are obtained sequentially, Garrigues et al [20] present a homotopy algorithm to compute the solution of the Lasso problem after receiving $n$ observations from the solution of the Lasso after receiving $n-1$ observations. This method is particularly efficient when the supports of the two solutions are close. Note that to address the issue of the nonsmoothness of the $l_1$ norm, most lasso algorithms optimize the dual of (1). For these algorithms, the solution computed with $n-1$ observations may not be used as a warm start to compute the solution with $n$ observations as it may no longer be feasible as we add new observations.

In this article, the vector $x^n$ is the state estimate of the system after receiving the $n^{th}$ observation. We are interested in sparse changes in the state vector as we receive new observations. To achieve this property, we add a $l_1$ penalty on the variation of the state vector, which regularizes the estimates when measurements are noisy and the dynamics of the system are slow compared to the sampling rate. We would like to re-solve the problem as we get a new observation using the information already available and without having to completely resolve the problem. This is akin to recursive least-squares, but now we have to handle the $l_1$-norm term added to the objective function. The estimation problem of $x^n$ is defined recursively as:

$$x^n = \arg\min_{x \in \mathbb{R}^m} \frac{1}{2} \sum_{i=1}^{n} (a_i^T x - y_i)^2 + \mu_n ||x - x^{n-1}||_1.$$
The algorithm is initialized assuming information $x_{\text{init}}$ on the state of the system (e.g. historical or previous estimate) and we set $x^0 = x_{\text{init}}$. When estimating $x^n$, we want the vector $x^n - x^{n-1}$ to be sparse. We call $x^{n-1}$ the reference parameter for the estimation of $x_n$. After receiving a new observation $(y_{n+1}, a_{n+1})$, we define a homotopy algorithm to update the estimate from $x^n$ to $x^{n+1}$ and vary the reference parameter from $x^{n-1}$ to $x^n$, before receiving the next data point.

In applications, it is useful to add additional regularization to the optimization problem (2). In particular, if we call $A$ the matrix whose $i^{th}$ row is equal to $a_i^T$, the matrix $A^T A$ should be non singular for the solution of the least-squares estimation problem to be unique. Moreover, the regularization term $\mu_n \|x - x^{n-1}\|_1$ is on the difference in successive parameters but there is no regularization to maintain the state estimates within given bounds (corresponding to physical characteristics of the system for example) or close to an apriori estimate of the state. We show how to leverage prior information $\hat{x}$ on the value of the parameter $x$ (from historical data for example) by adding a $l_2$ regularization term to problem (2), with weighting parameter $\lambda$. We also propose an algorithm that ensures that the estimates remain within given upper and lower bounds $\bar{x}$ and $\underline{x}$. The resulting estimation scheme amounts to solving the following optimization problem:

$$\text{minimize} \quad \frac{1}{2} \sum_{i=1}^n (a_i^T x - y_i)^2 + \mu_n \|x - x^{n-1}\|_1 + \frac{\lambda}{2} \|x - \hat{x}\|_2^2$$

s.t. \[ \underline{x} \leq x \leq \bar{x} \quad (3) \]

where the inequalities $\underline{x} \leq x$ and $x \leq \bar{x}$ refer to componentwise inequalities on the vectors. This article presents an online estimation algorithm of a time varying system with sparse changes on the state vector between successive estimations. The algorithm is based on homotopy algorithms for the variation of the regularization parameter $\mu_n$ [16], [17], [18] and the addition of a new observation as it becomes available [20]. The contribution of this article is the presentation of a homotopy algorithm to produce sparse variations in the state estimate and in particular, update the reference parameter each time a new observation is received. This algorithm is particularly efficient to solve large scale online estimation problems, as demonstrated in this article. We present the estimation algorithm with an additional $l_2$-regularization and bounds on the state vector, necessary to regularize the state estimates and ensure that the results meet physical requirements of the system.

This article is organized as follows. We review the optimality conditions of the Lasso algorithm (Section II) and present an algorithm to compute a homotopy on the parameter vector $x$ solving the optimization problem (3) (Section III). We apply these results to traffic estimation on an arterial network in San Francisco, CA (Section IV). We estimate the mean travel time on each link of the network, totalling more than 12.6 kilometers of roadway, from GPS data sent by 500 probe vehicles sampled every minute (Figure 1). The data is collected by the Mobile Millennium system [21] which receives on the order of 500,000 points daily in the San Francisco Bay Area. We discuss possible extensions of this work in Section V.

II. Optimality Conditions for the Lasso

The objective function in (1) is convex and non-smooth since the $l_1$-norm is not differentiable when there exists $i$ such that the $i^{th}$ element of $x$ (denoted $x_i$) equals zero. Hence there is a global minimum at $x$ if and only if the subdifferential of the objective function at $x$ contains the 0-vector. The subdifferential of the $l_1$-norm at $x$ is the following set

$$\partial \|x\|_1 = \left\{ v \in \mathbb{R}^m : \begin{array}{ll} v_i = \text{sgn}(x_i) & \text{if } |x_i| > 0 \\ v_i \in [-1, 1] & \text{if } x_i = 0 \end{array} \right\}$$

where $\text{sgn}(\cdot)$ is the sign function. Let $A \in \mathbb{R}^{n \times m}$ be the matrix whose $i^{th}$ row is equal to $a_i^T$, and let $y = (y_1, \ldots, y_n)^T$ be the vector of response variables. The optimality conditions for the Lasso problem (1) are given by $A^T (Ax - y) + \lambda v = 0$, $v \in \partial \|x\|_1$.

We define as the active set (resp. non active set), the indices representing non-zero elements (resp. zero elements) of $x$. The active and non-active sets are referenced by the subscripts 1 and 2 respectively. For example $A_1$ is the matrix representing the columns of $A$ in the active set. The vector $x_1$ represents the non-zero coordinates of $x$ and $x_2$ is the 0-vector. The index 1 (resp. 2) references the $i^{th}$ coordinate of the active (resp. non active set). Since $v \in \partial \|x\|_1$, the $i^{th}$ coordinate of $v_1$ is $v_1 = \text{sgn}(x_{i1})$, and the $i^{th}$ coordinate of $v_2$ is $v_2 \in [-1, 1]$. If the solution is unique, it can be shown that $A_1^T A_1$ is invertible, and we can rewrite the optimality conditions as

$$x_1^* = (A_1^T A_1)^{-1} (A_1^T y - \mu_0 v_1) - \mu_0 v_2 = A_2^T (A_2 x_1^* - y)$$

Note that if we know the active set and the signs of the coefficients of the solution, thus the vector $v_1$, we can compute it in closed form. When solving the estimation problem (2) or (3), we define a change of variable $x_\phi = \phi(x)$ such that if $x$ is the solution of the estimation problem, $x_\phi$ is typically sparse. We present an algorithm to update the active set and the signs of the coefficients of the solution.
$x_r$ from previous solutions by computing a continuous path between the successive state estimates.

### III. Homotopy Algorithm

Suppose that we have computed the solution $x^n$ to Equation (3) with $n$ observations. The reference parameter is $x^{n-1}$ since we would like the variations between $x^{n-1}$ and $x^n$ to be sparse. We receive an additional observation $(y_{n+1}, a_{n+1}) \in \mathbb{R} \times \mathbb{R}^m$ and a new penalty coefficient $\mu_{n+1}$. In general, $\mu_n$ reflects the number of measurements. We also need to update the reference parameter from $x^{n-1}$ to $x^n$. We introduce the following optimization problem:

$$
\min_{x_r \in \mathbb{R}^n} \frac{1}{2} ||Ax_r - y_r - A_1 x_{rc}||^2 + \frac{\mu}{2} ||x_r - (\hat{x} - x^{n-1})||^2
$$

s.t. $x_r \leq \bar{x}_r \leq \check{x}_r$

We write the optimality conditions for each set of indices (active indices, active constraints, non active indices):

$$
\begin{align*}
(A_1^T A_1 + \lambda I)s_{1,1}^*(t) - b_1 + (r^2 - 1)a_{n+1}^T (a_{n+1}^T s_{1,1}^*(t) - y_{n+1}) + \mu_{n+1} v_{n+1} &= 0 \\
E_c (A_1^T A_1 s_{1,1}^*(t) - b_1 + (r^2 - 1)a_{n+1}^T (a_{n+1}^T s_{1,1}^*(t) - y_{n+1}) + \mu_{n+1} v_{n+1}) &\geq 0 \\
A_1^T A_1 s_{1,1}^*(t) - b_2 + (r^2 - 1)a_{n+1}^T (a_{n+1}^T s_{1,1}^*(t) - y_{n+1}) + \mu_{n+1} v_{n+1} &= 0
\end{align*}
$$

With this change of variable, the derivation of $x_r^e(t)$ is similar to the calculation performed in [20]. We use the Sherman-Morrison formula and solve for $x_r^e(t)$:

$$
x_r^e(t) = \frac{(r^2 - 1) \tilde{e}}{1 + \alpha (r^2 - 1)} q
$$

where $\tilde{e} = A_1 x_{rc} - y_r$. We replace these expressions in the optimality conditions for the active constraint indices $c$ and the non active indices $2$. Let $t_c$ be the value of $t$ that sets the $i^{th}$ coordinate of $x_r^e(t)$ to zero.

$$
t_c = \left(1 + \left(\frac{\tilde{e}_i}{x_{r,t_c} - \check{x}_{r,t_c}} - \alpha \right)^{-1}\right)^{1/2}
$$

We denote by $t_0$ (resp. $t_2^-$), the value of $t$ that sets the parameter $x_r^e$ to $\check{x}_{r,1}$ (resp. $x_r^e$).

$$
t_0 = \left(1 + \left(\frac{\tilde{e}_i}{x_{r,t_0} - \check{x}_{r,t_0}} - \alpha \right)^{-1}\right)^{1/2}
$$

and have a same expression for $t_2^-$, replacing $\check{x}_r$ by $\underline{x}_r$. We notice that

$$
\begin{align*}
A_1^T x_r^e(t) - y_r &= \tilde{e} - \frac{1}{1 + \alpha (r^2 - 1)} A_1 q \\
A_1^T x_r^e(t) - y_r &= \tilde{e} - \frac{1}{1 + \alpha (r^2 - 1)} A_1 q
\end{align*}
$$

We introduce a change of variables to formulate the optimization problem as a Lasso problem and solve it using an online Lasso algorithm [20]. We define $x_r = x - x^{n-1}$ and solve the optimization problem for $x_r$. The vector $x_r$ must satisfy coordinate by coordinate inequalities $x_r \leq x_r \leq \bar{x}_r$, with $x_r = \hat{x} - x^{n-1}$ and $\bar{x}_r = x - x^{n-1}$. If the $i^{th}$ element of $x_r$ equals $x_{r,i}$ (resp. $x_r$), we say that the upper (resp. lower) bound $i$ is an active constraint and reference the set of active constraints by the subscript $c$. We reference the $i^{th}$ coordinate of the active constraints by the index $c_i$ and define $e_{c_i} = -1$ (resp. $e_{c_i} = 1$) if the upper (resp. lower) bound is active. We denote by $E_c$ the diagonal matrix with $i^{th}$ diagonal element equal to $e_{c_i}$.

The matrix $A$ and the vector $y$ represent $n + 1$ observations of the state. Given the set of active constraints $c$, we define $y_r = y - Ax^{n-1} - A_1 x_{rc}$. We define $b_1 = A_1^T y_r + \hat{\lambda} [\hat{x} - x^{n-1}]$, where $j$ represents the set of indices 1 or 2 (active and non-active). We define $b_c = A_1^T y_r + \hat{\lambda} [\hat{x} - x^{n-1}] - \lambda x_{rc}$ and $K = (A_1^T A_1 + \hat{\lambda} I)^{-1}$. We note $A_{n+1}^T$ the last row of $A$ and $y_{r,n+1}$ the last element of $y_r$, which correspond to the last observation received. Let $x_r^e(t)$ be the solution of the following optimization problem:

$$
\min_{x_r \in \mathbb{R}^n} \frac{1}{2} ||Ax_r - y_r - A_1 x_{rc}||^2 + \frac{\mu}{2} ||x_r - (\hat{x} - x^{n-1})||^2
$$

s.t. $x_r \leq \bar{x}_r \leq \check{x}_r$

We write the optimality conditions for each set of indices (active indices, active constraints, non active indices):

$$
\begin{align*}
(A_1^T A_1 + \lambda I)s_{1,1}^*(t) - b_1 + (r^2 - 1)a_{n+1}^T (a_{n+1}^T s_{1,1}^*(t) - y_{n+1}) + \mu_{n+1} v_{n+1} &= 0 \\
E_c (A_1^T A_1 s_{1,1}^*(t) - b_1 + (r^2 - 1)a_{n+1}^T (a_{n+1}^T s_{1,1}^*(t) - y_{n+1}) + \mu_{n+1} v_{n+1}) &\geq 0 \\
A_1^T A_1 s_{1,1}^*(t) - b_2 + (r^2 - 1)a_{n+1}^T (a_{n+1}^T s_{1,1}^*(t) - y_{n+1}) + \mu_{n+1} v_{n+1} &= 0
\end{align*}
$$

We introduce a change of variables to formulate the optimization problem as a Lasso problem and solve it using an online Lasso algorithm [20].
These derivations allow the computation of a continuous update of the state as a new observation is received.

B. Step 2: varying the regularization parameter

We use the change of variables from Section III-A and solve the Lasso problem where we vary the penalization parameter [16], [17], [18]. With a slight abuse of notation on arguments (but for notational simplicity), we denote by \( x_r^*(\mu) \) the solution of the optimization problem

\[
\min_{x_r \in \mathbb{R}^m} \frac{1}{2} ||Ax_r - y_r - A_x x_r||_2^2 + \mu ||x_r||_1 + \frac{\lambda}{2} ||x_r - (\hat{x} - x^{a-1})||_2^2.
\]

\[\text{s.t.} \quad x_r \leq x_r \leq x_r\]

The information on the active set and the signs of the parameters in the active set is available at \( \mu = \mu_r \) (Step 1). The active set, active constraints and signs remain constant for \( \mu \) in an interval \([\mu_r, \mu^*]\) where the solution \( x_r^*(\mu) \) is affine in \( \mu \). As we reach the "transition point" \( \mu^* \), we update the active set, active constraints and signs which remain valid until the next transition point. The optimality conditions read:

\[
(A^T_A + \lambda I)x_r^*(\mu) - b_1 + \mu v_1 = 0 \quad \text{and} \quad (A^T_A x_r^*(\mu) - b_c + \mu v_c) \geq 0.
\]

Solving for \( x_r^*(\mu) \), we have \( x_r^*(\mu) = K(b_1 - \mu v_1) \). Denoting by \( \mu_1 \) the value of \( \mu \) that sets the \( i \)-th coordinate of \( x_r^* \) to zero, we read \( \mu_1 = [Kb_1]_i / [Kv_1]_i \).

The upper bound (resp. lower bound) becomes active as \( x_r^*(\mu) \) equals \( x_r^*(\mu) \) (resp. \( x_r^*(\mu) \)), when \( \mu \) equals \( \mu_1 \) (resp. \( \mu_2 \)), i.e. \( \mu_2 = [Kb_1 - x_r^*(\mu)] / [Kv_1]_i \).

Let \( \mu_i \) be the value of \( \mu \) such that the \( i \)-th component of \( x_r^* \) is no longer an active constraint. Let \( \mu_1 \) be the value of \( \mu \) such that \( w_2(\mu) = 1 \) (resp. \( w_2(\mu) = -1 \)). From the expression of \( x_r^*(\mu) \) and the optimality conditions, we see that the partial derivatives of the objective function for coordinates in \( c \) and the function \( \mu \rightarrow w_2(\mu) \) are affine in \( \mu \). We derive the expressions of \( \mu_i, \mu_2 \) and \( \mu_2 \) as

\[
\mu_i = (b_i - A^T_c A_1 K b_1) / (v_i - A^T_c A_1 K v_1)
\]

\[
\mu_2 = (b_2 - A^T_c A_1 K b_1) / (1 - A^T_c A_1 K v_1)
\]

C. Step 3: Updating the reference parameter

After adding the new observation and varying the regularization parameter, the algorithm updates the reference parameter from \( x^{a-1} \) to \( x^a \). We define \( \Delta x = x^a - x^{a-1} \) and the change of variable \( x_r^a(u) = x^a - [1 - u]x^{a-1} + u x^u \), which is the vector that we impose sparsity on. The constraints on this variable are \( x_r^a(u) = x - [(1 - u)x^{a-1} + u x^u] \), which is the vector that we impose sparsity on. The constraints on this variable are \( x_r^a(u) = x - [(1 - u)x^{a-1} + u x^u] \) and \( x_r^a(u) \) (defined similarly from \( \mathbb{G} \)). Given a value of \( u \) and a set of active constraints \( c \), we define \( x_r^0 = x_r(u) + u (\Delta x) \), and notice that this vector no longer depends on \( u \). The \( i \)-th element of this vector is equal to \( x_r^0_i - x_r^{a-1}_i \) (resp. \( x_r^0_i - x_r^{a-1}_i \)) if the lower (resp. upper) bound is active. We define \( y_r = y - Ax^{a-1} - Ax_r^0 \).

\[\text{Algorithm 1 Homotopy algorithm for online state estimation of time varying systems with sparse temporal changes.}\]

1. Add the latest observation \((s_{n+1}, y_{n+1})\): Compute the path from \( x^a = x(0, 0, \mu_a) \) to \( x(1, 0, \mu_{a+1}) \). Refer to the derivations of this article and to [16], [17], [18] for the details of the algorithm.

2. Vary the regularization parameter: Compute the path from \( x(1, 0, \mu_a) \) to \( x(1, 0, \mu_{a+1}) \). Refer to the derivations of this article and to [20] for the details of the algorithm.

3. Initialize the active set to the non-zero coefficients of \( x_r(0) = x(1, 0, \mu_{a+1}) - x^{a-1} \). Initialize \( v_1 = \text{sgn}(x_r^a(0)) \), \( u^r = 0 \) and \( K = (A^T_A + \lambda I)^{-1} \).

4. Compute the next transition point \( u^r \). If it is smaller than the previous transition point or greater than 1, go to Step 6. Otherwise:

a. The \( i \)-th component of \( x_r^a(u^r) \) goes to zero: remove \( i \) from the active set.

b. The \( i \)-th component of \( x_r^a(u^r) \) reaches \( x_r(u) \) or \( x_r(u) \): remove \( i \) from the active set and add it to the active constraints.

c. The \( i \)-th component of \( w_2(u^r) \) reaches one in absolute value: add \( i \) to the active set.

d. The \( i \)-th optimality condition of the active constraints reaches zero: add \( i \) to the active set.

5. Update \( v_1 \) and \( A_1 \) according to the updated active set and sign of the parameters. Update \( K = (A^T_A + \lambda I)^{-1} \), (rank 1 update). Go to Step 4.

6. Compute the final value of \( x_r^a(1) \).
derivative of the objective function equals zero:
\[ A_1^T A_1 \xi - b_1 + \mu v_c + u_c \left( A_1^T (A_1 \chi + A \Delta x - A_c (\Delta x)_c) \right) = 0, \]
from which we read the expression of \( u_{c_i} \).

We solve for \( u_{2_i}^+ \) (resp. \( u_{2_i}^- \)), where \( u_{2_i}^+ \) (resp. \( u_{2_i}^- \)) is the value of \( u \) for which the \( i^{th} \) component of \( x_{i,2} \) enters the active set and becomes positive (resp. negative). They are given by:
\[
\begin{align*}
    u_{2_i}^+ &= -\frac{A_1^T A_1 \xi - b_2 + \mu}{A_2^T (A_1 \chi + A \Delta x - A_c (\Delta x)_c) + \lambda (\Delta x)_{2_i}} \\
    u_{2_i}^- &= -\frac{A_1^T A_1 \xi - b_2 - \mu}{A_2^T (A_1 \chi + A \Delta x - A_c (\Delta x)_c) + \lambda (\Delta x)_{2_i}}
\end{align*}
\]

### IV. EXPERIMENTAL RESULTS

We apply the algorithm to arterial traffic estimation on a subnetwork of San Francisco, CA consisting in 815 links (Figure 2). Arterial traffic is modeled as a time varying system and we seek to estimate travel times on each link of the network, as they vary over time. Traffic data on arterial networks is mainly provided from probes sending their location at a given sampling frequency (common sampling frequencies are around 1 minute). The proportion of sampled vehicles (penetration rate) remains limited and rarely exceeds a few percent of the vehicles traveling on the network. Moreover, traffic signals cause important variation on the travel time experienced on a link of the network within very short periods of time (depending on whether the vehicle stopped at the signal or not), while the actual changes in traffic conditions have slower dynamics. Given the penetration rate of probe vehicles, we seek to estimate trends in traffic conditions rather than fluctuations around a mean value. For these reasons, arterial traffic estimation is a good application for the algorithm. The parameter \( x^n \) represents the average travel time on each link after receiving the \( n^{th} \) observation and we impose sparsity on its temporal evolution.

Fig. 2. Subnetwork of San Francisco used for arterial traffic estimation.

Fig. 3. Paths of three probe vehicles on a network with eleven links. The path of a probe is represented as a vector \( a_i \in [0,1]^m \) where the \( j^{th} \) coordinate of \( a_i \) represent the fraction of link \( j \) traveled by the probe. The path represented with a solid line is represented with a sparse vector with non zero coordinates 1, 6 and 9, respectively equal to 0.4, 0.7 and 1 considering that the probe traveled 40% of link 1 and 70% of link 6. The vector representing the dashed path has non zero coordinates 2, 3, 8 and 11, respectively equal to 0.3, 1, 0.8 and 1 considering that the probe traveled 30% of link 2 and 80% of link 8.

**Experimental setup:** Beginning in March of 2009, data has been collected from probe vehicles in the San Francisco Bay Area by the Mobile Millennium system [21]. A fleet of over 500 taxis report their location every minute, along with an identifier and a status (carrying a passenger or not) [22] allowing filtering of the taxi stops to load or unload passengers. The duration between two successive location reports \( z_1 \) and \( z_2 \) represents an observation of the travel time \( y \) of the vehicle on its path from \( z_1 \) to \( z_2 \). We use an algorithm [23] that combines models of GPS measurements and drivers’ behavior into a conditional random field to provide trajectory reconstructions between \( z_1 \) and \( z_2 \). The latency in the communication of the location data to our servers is generally less than a few minutes.

Each trajectory (path) is converted in a vector \( a_i \in [0,1]^m \), where \( m \) is the number of links in the arterial network. The \( k^{th} \) coordinate of \( a_i \), \( a_{i,k} \) is the fraction of the link traveled by the probe vehicle, computed as the distance traveled on the link divided by the length of the link. In particular, \( a_{i,k} = 0 \) if the vehicle did not travel on link \( k \) and \( a_{i,k} = 1 \) if the vehicle fully traversed the link \( k \) (see Figure 3). Note that on arterial links the mean travel time on a fraction of the link does not vary proportionally with the distance traveled. Vehicles are more likely to experience delays close to the downstream intersection because of the presence of traffic signals [24], [25], and thus the coefficients \( a_{i,k} \) should take into account the locations where the vehicles started and ended their travel on link \( k \). However, these considerations are not taken into account in this article.

**Numerical experiments:** We learn the parameter \( \lambda \), solving equation (3), which represents the mean travel time on each link of the network. We initialize the algorithm using a previous estimate of the mean travel times given by least-squares regression and use historical mean travel times for the \( l_2 \) regularization \( \lambda \). Each time we receive a new travel time observation, we add the new observation \( (y_{n+1}, a_{n+1}) \in \mathbb{R} \times \mathbb{R}^m \) (Section III-A), increase the regularization parameter from \( n\mu \) to \( (n+1)\mu \) (Section III-B) and
update the reference parameter (Section III-C). As we are interested in the estimation of the current state of the system, observations of the state may only be relevant for a limited period of time. We consider that each observation remain relevant for a time $T$ after being received in the system. When an observation $(y_j, a_j) \in \mathbb{R} \times \mathbb{R}^m$ becomes obsolete, we (1) remove the old observation $(y_j, a_j) \in \mathbb{R} \times \mathbb{R}^m$ by decreasing $t$ from 1 to 0, (2) decrease the regularization parameter from $(n+1)\mu$ to $n\mu$, where we assume that $n$ represents the number of observations currently considered relevant for the estimation and (3) update the reference parameter.

We want to assess the performance of the model and quantify the effect of the regularization parameters $\lambda$ and $\mu$. The first parameter penalizes solutions which are far (in the $l_2$-norm sense) from the historical estimate of travel times $\hat{x}$. The second parameter imposes sparsity on the variation of the state. The choice of these parameters leads to a compromise between (1) fitting the data, with risks of overfitting and lack of physical interpretation and (2) putting too much weight on the regularization and not estimating accurately the current state of the system.

In this case study, we estimate traffic conditions using taxi data collected on September 3, 2010 from 5:00pm to 7:00pm in a subnetwork of San Francisco. This subnetwork contains 815 links (where a link is defined as the road between two signals) totalling more than 12.6 kilometers of roadway.

We use cross-validation to assess the performance of our model, randomly splitting the observations sent by the probe vehicles between a training set and a validation set. After learning the travel time estimates on the training set, we use the validation set to compare our estimates to the travel time observations. We compare the performance of our model with a baseline model, which uses the historical value of the link travel times $\hat{x}$ as the estimate of the state. We consider three validation metrics which, even though closely related, give different information on the quality of the estimation: the root mean squared error (RMSE), the mean absolute error (MAE) and the mean percentage error (MPE). The algorithm minimizes the $l_2$ norm of the residual between the estimate and the observation. The RMSE indicates the goodness of fit with validation data. Note that the variability of arterial travel times (due to traffic signals, pedestrians, etc.) leads to important fluctuations of travel times. This inherent variability in the state of the system makes our estimation model robust with sparse variations, but is also responsible for relatively high values of the error metrics. For example, the RMSE is greater than the standard deviation of travel times [24].

The results indicate that both the $l_1$ and the $l_2$ regularization (Figure 4) are important to improve the estimation capabilities. For a wide range of parameters, the results are

\[ \text{RMSE} = \sqrt{\frac{\sum_{o=1}^{N} (y_o - \hat{y}_o)^2}{N}} \quad \text{MAE} = \frac{\sum_{o=1}^{N} |y_o - \hat{y}_o|}{N} \quad \text{MPE} = \frac{1}{N} \sum_{o=1}^{N} \frac{|y_o - \hat{y}_o|}{y_o} \]
significantly better than the baseline model. The results also underline the importance of the additional $l_2$ regularization to improve the robustness of the algorithm. Figure 5 illustrates that in addition to improving the estimation capabilities, the algorithm produces results that are easier to interpret. Arterial traffic is highly variable and the variability often prevents the interpretation of the results. With this model, we are able to deal with this variability in order to estimate the trends in travel times on the links of the network.

V. CONCLUSION AND POSSIBLE EXTENSIONS

This article derives an algorithm for an online least-squares estimation of the state $x$ of a time-varying system from successive observations $(y_t, a_t) \in \mathbb{R} \times \mathbb{R}^m$. We use $l_1$-norm regularization to limit the variations in the estimate of the state to capture the trends in the dynamics rather than the fluctuations. We add $l_2$-norm regularization to increase the robustness of the estimator and limit numerical issues when the matrix $A^T A$ is singular (or ill-conditioned), where $A$ is the matrix with line $i$ equal to $a_i^T$. Constraints ensure that the state estimates remain within feasible bounds.

The homotopy algorithm is particularly efficient when the variations between the estimates are sparse, leading to few transition points. The algorithm computes a continuous path, which is in general not possible for other lasso algorithms which solve the dual problem. Moreover, the computational costs are limited as all matrix inverses are computed with rank 1 updates. The number of transition points and active indices varies with the parameter $\mu$. As $\mu$ increases, the number of transition points and active indices decreases, improving the computational efficiency of the algorithm. For small values of $\mu$, the algorithm may not be as efficient, as the number of transitions is bounded by $3^m$.

The model provides a significant improvement in the estimation capabilities, compared to a baseline model of arterial traffic estimation with probe data. We achieve sparse variations in the parameter and estimate the global trends in traffic conditions by filtering out the noise due to fluctuation. The number of transition points and active indices remain small throughout the algorithm (inferior to ten for a network with 815 links). This algorithm could be developed further to study change detection in time-varying systems. We are also investigating generalization of this algorithm to more general forms of $l_1$-norm regularizations. For example, we could be interested in sparse spatial variations of the estimates.

ACKNOWLEDGMENTS

The authors wish to thank Timothy Hunter and Ryan Herring from UC Berkeley for providing filtered probe trajectories from the raw measurements of the probe vehicles as well as valuable visualization capabilities of the data. We also thank Ryan Herring for his help in proofreading. We thank the members of the staff of the California Institute for Innovative Transportation for their contributions to develop, build, and deploy the system infrastructure of Mobile Millennium on which the numerical experiments of this article rely.

REFERENCES