GOSUS: Grassmannian Online Subspace Updates with Structured-sparsity

Jia Xu[†], Vamsi K. Ithapu[†], Lopamudra Mukherjee[§], James M. Rehg[‡], Vikas Singh[†]
[†]University of Wisconsin-Madison, [§]University of Wisconsin-Whitewater
[‡]Georgia Institute of Technology
http://pages.cs.wisc.edu/~jiaxu/projects/gosus/

Abstract

*We study the problem of online subspace learning in the context of sequential observations involving structured perturbations. In online subspace learning, the observations are an unknown mixture of two components presented to the model sequentially — the main effect which pertains to the subspace and a residual/error term. If no additional requirement is imposed on the residual, it often corresponds to noise terms in the signal which were unaccounted for by the main effect. To remedy this, one may impose 'structural' contiguity, which has the intended effect of leveraging the secondary terms as a covariate that helps the estimation of the subspace itself, instead of merely serving as a noise residual. We show that the corresponding online estimation procedure can be written as an approximate optimization process on a Grassmannian. We propose an efficient numerical solution, GOSUS, **G**rassmannian **O**nline **S**ubspace **U**pdates with **S**tructured-sparsity, for this problem. GOSUS is expressive enough in modeling both homogeneous perturbations of the subspace and structural contiguities of outliers, and after certain manipulations, solvable via an alternating direction method of multipliers (ADMM). We evaluate the empirical performance of this algorithm on two problems of interest: online background subtraction and online multiple face tracking, and demonstrate that it achieves competitive performance with the state-of-the-art in near real time.*

1. Introduction

Subspace learning methods have been extensively studied in vision with applications spanning motion analysis, clustering, background estimation, and deriving semantic representations of scenes [11, 7, 6, 13]. Within the last few years, new developments in matrix factorization [36, 3] and sparse modeling [25, 38] have led to significant renewed interest in this construct, and has provided a suite of new models and optimization schemes for many variants of the problem. An interesting version that several authors have proposed recently is *Online Subspace Learning* [37, 4, 15]. Here, observations are presented sequentially, in the form of an unknown mixture of the primary subspace(s) plus a residual component. The objective is to keep an estimate of the contributing subspace(s) updated as the observations continually present themselves.

The standard strategy of modeling the foregoing online estimation question is to assume that the observation is an unknown mixture of two components. The first relates to the *subspace* terms comprising one or multiple subspaces (and with or without regularization). Statistically, one may regard this term as the *main effect* which explains *most* of the measurement. But fitting the signal to high fidelity will necessarily involve a large degree of freedom in the subspace term, and so the model allows for a small amount of compensatory residual error — this corresponds to the second term contributing to the observed signal. To encourage the residual quantity to be small, most proposals impose a sparsity penalty on its norm [24, 15]. Therefore, the main technical concern, both in the "batch" and online settings, is to efficiently estimate the subspace and if possible provide probabilistic guarantees of correct recovery.

Within the last year, a particularly relevant application of online subspace learning is in the context of keeping updated estimates of background and foreground layers for video data[1]. Here, one exploits concepts from matrix completion for subspace estimation, by drawing i.i.d. samples from each incoming frame, and adjusting the current subspace parameters using *only* the sub-sampled data [15]. The mass of the signal outside the support of the subspace may then be labeled as foreground. This strategy works quite well when the background is completely static: essentially, the model has seen several hundred frames and has converged to a good estimate already. However, when there are small but continual variations in the background (e.g., a swaying tree) and/or it is undergoing changes due to camera motion, zoom or illumination differences, it takes time

[1]We will use this as a running example throughout the paper in an effort to make certain ideas concrete (we present results for another application in Section 5.2, thereby demonstrating the generality of the method).

1550-5499 2013
DOI 10.1109/ICCV.2013.419

for the subspace estimates to stabilize. Here, the residual must then compensate for a less than ideal estimate of the main effect, which leads to salt-pepper isolated foreground regions, scattered over the image. One reason for this unsatisfactory behavior is that the model does not enforce spatial homogeneity in the foreground region. Imposing 'structure' on the secondary term, such as asking for contiguity, has the highly beneficial effect that the residual serves a more important role than merely accounting for the error/corruption. From a statistical modeling perspective, the residual structure acts as a covariate that improves the estimate of the main effect (the background reconstruction via subspace modeling). Consequently, in the background/foreground setting, we see that the estimated foreground regions are far more meaningful. The resultant improvements in performance are quite significant, compared to the alternative. For several other interesting applications which we discuss later in the paper, the benefits are clear, though the notion of structure (i.e., structured sparsity operator) is different and better reflects the needs of that domain.

This paper. Consider a regression model, $Y = f(W) + \epsilon$. If the distributional properties of the second term is known (e.g., Rician, Poisson), it must improve the estimation of $f(\cdot)$. We seek to translate this simple idea to the problem of Online Subspace Learning, by incorporating structure (i.e., via a group norm) on the secondary term. The **key** contributions of this paper are: **1)** Show how group sparsity based structural homogeneity can be incorporated within estimation problems defined on Grassmannian manifolds; **2)** Present an efficient online optimization scheme where most constituent steps reduce to simple matrix operations; **3)** Demonstrate for two example applications (online background subtraction and online multiple face tracking) using a variety of datasets, that the method gives competitive empirical performance in near real time.

2. Related Work

Subspace learning, and more generally, learning low dimensional multi-linear models has a long and rich history in Computer Vision. The contemporary suite of algorithms for this problem may be classified into a few separate categories, which nonetheless share important similarities. Models inspired from dimensionality reduction techniques build upon the traditional principal component analysis (PCA) framework. For instance, *Robust subspace learning* [11, 13] and *Generalized Principal Component Analysis* (GPCA) [34] take a hybrid geometric/statistical view of separating heterogeneous 'mixed' data drawn from one or more subspaces. Building upon classical approaches based on factor analysis, independent component analysis (ICA) and its variants [23] parameterize the subspace as a combination of a small set of sources [18], and work well for subspace estimation applications such as action recognition [21], segmentation [27] and facial pose analysis [23]. More recently, theory from compressive sensing (also, matrix completion) [9], and matrix factorization [3] have been successfully translated into new models and optimization schemes for this problem. An important representative from this group, which has found a multitude of vision applications, is Robust Principal Components Analysis (RPCA) which expresses the measurement as a combination of a low rank matrix and a ℓ_1-regularized noise component [24, 8]. Separately, several authors express subspace estimation as a non-negative matrix factorization (NMF) [36, 6, 3] and give rigorous recovery guarantees. While the literature devoted to the batch setting above is interesting, there is also brisk research activity in vision, especially in the last two years, focused on the *online* version of this problem. This has led to a set of powerful online subspace learning methods [37, 4, 15], which are related to the above ideas as well as a parallel body of work in manifold learning [14, 32] — they leverage the fact that the to-be-estimated signal lies on a Grassmannian [32]. In particular, GROUSE [4] and GRASTA [15] (an online variant of RPCA) show how the subspace updates can be accurately maintained in real time by using sub-sampling ideas. Our framework leverages this body of work, and we will point out similarities to known results in the presentation that follows.

3. Model design

Notations. We denote matrices by non–bold upper case letters (e.g. V), vectors by bold lower case (e.g. $\mathbf{x}$) and scalars by non-bold lower case letters (e.g., μ). Subscripts and superscripts will denote frame numbers, iterations, indices, etc., which will be explained as needed.

This section describes the various sub-components that make up the main model studied in this paper. As introduced in Section 1, the data V is a composition of a main effect (or signal) B and a secondary term (or outlier) X. That is, $V = B + X$ where $V, B, X \in \mathbb{R}^{n \times m}$, n is the data dimensionality and m is the number of observations. The signal B is given as a linear combination of d sources (subspace basis) in n dimensions, denoted by $U = [\mathbf{u}_d]$. This assumption is reasonable since the variation in signal across consecutive frames is small enough that it allows the few ($d \ll n$) degrees of freedom to recover most changes. The orthogonal structure of U implies that it lies on a Grassmannian manifold $\mathcal{G}_{n,d}$ embedded in a n-dimensional Euclidean space. Let the coefficient matrix be W. In the absence of any error, we have $B = UW$. Now, if $\mathbf{v} \in \mathbb{R}^n$ is an observation and $\mathbf{x} \in \mathbb{R}^n$ is the corresponding outlier vector (lies outside the support of the subspace given by U), then $\mathbf{v} = U\mathbf{w} + \mathbf{x}$, where $\mathbf{w}$ is the coefficients vector for the current observation. This expression is under constrained when both the signal and the outlier are unknown. To drive the estimation procedure, we impose a regularization constraint

expressing what constitutes a 'good' outlier, for instance, contiguity. That is, we may ask that the outlier be spatial coherent ensuring that isolated detections scattered across the image are strongly discouraged. The implicit expectation is that this makes $\mathbf{x}$ more meaningful in the context of the application, and so usefully biases the estimation of the subspace. We elaborate on the notion of structure next.

3.1. Structured sparsity

For the background estimation example, the texture/color of the foreground objects (i.e., outliers) is homogeneous and so the outliers should be contiguous in an image. For multiple face tracking (which we elaborate later), we need to *track* a set of faces in a given video where the subspace constitutes the faces themselves. But the outliers created by occlusions are *not* pixel sparse, instead, constitute contiguous regions distributed at different face positions [19]. As an example, consider a person wearing sunglasses or if a shadow or irregular illumination is distorting a part of the face. We do not want such occlusions to cause large changes in the online updates and destroy the notion of a face subspace. Instead, we must allow the $\mathbf{x}$ term to subsume and accommodate such structured deviations from a 'face' subspace.

To formalize this prior on the outlier, we use structured (or group) sparsity [39, 16, 17]. For one image frame, the groups may correspond to sets of sliding windows on the image, super-pixels generated via a pre-processing method (which encourages perceptually meaningful groups), or potential face sub-regions. A $n \times n$ (n is the dimensionality of each observation) diagonal matrix D^i is used to denote a "group" i. Each diagonal element of D^i corresponds to the presence/absence of a pixel in the i^{th} group, as

$$D^i_{jj} = \begin{cases} 1 & \text{if pixel } j \text{ is in group } i; \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

where D^i_{jj} is the j^{th} diagonal element of D^i. A penalty function is then defined as,

$$h(\mathbf{x}) = \sum_{i=1}^{l} \mu_i \|D^i \mathbf{x}\| \tag{2}$$

where μ_i gives the weight for group i and l is the number of such groups. D^i is sparse and allows overlap with other D^js ($i \neq j$), so that we can form groups from overlapping homogeneous regions (groups may also be disjoint, if desired). Our group sparsity function $h(\cdot)$ in (2) has a mixed norm structure. The inner norm is either l_2 or l_∞ (we use l_2) forcing pixels in the corresponding group to be similarly weighted, and the outer norm is l_1 which encourages sparsity (i.e. only few groups are selected). In general, the design of D^is depends on the needs of the application. We will give specific examples shortly.

3.2. Model

With these components in hand, we can now present our main model. Given an input data $V \in \mathbb{R}^{n \times m}$, our model estimates the subspace matrix U, the coefficient vector $\mathbf{w}$, and the outlier $\mathbf{x}$, at a given time point (where $\mathbf{v}$ denotes the given current observation) by the following minimization, (λ is a positive regularization parameter)

$$\min_{U^T U = I_d, \mathbf{w}, \mathbf{x}} \quad \sum_{i=1}^{l} \mu_i \|D^i \mathbf{x}\|_2 + \frac{\lambda}{2} \|U\mathbf{w} + \mathbf{x} - \mathbf{v}\|_2^2 \tag{3}$$

4. Optimization

While model (3) faithfully models our requirements, optimizing it can be challenging. This is due to the non-smoothness of the mixed norm and non-convexity arising due to the orthogonal structure of U. In fact, several recent papers [26, 10, 29] are devoted to ideas for optimizing the structured sparsity norm objectives *alone*, and even by itself, it gets complicated due to overlapping groups. Specifically, one may require the design of specialized proximal operators, and the running time of many existing schemes ($\sim$ 30 minutes, [29]) is impractical for problem sizes encountered in our application.

Observe that at any given time point, the model has already processed many frames before it, and has obtained a reasonable estimate of U. Because the changes in U are not drastic from one frame to the other, local updates of the variables in (3) are sufficient in practice. This is a compromise since obtaining a global optimum for the nonconvex U is unlikely anyway. We adopt a block-wise approach which solves for a subset of variables keeping the others fixed. In particular, we observe (3) is convex for $(\mathbf{w}, \mathbf{x})$ when U is fixed, which can be computed efficiently. A sequential update scheme [28] is used when optimizing for U, while still preserving its orthogonality. Below, we give a detailed analysis of these sub-procedures and outline methods to optimize each component and the overall model.

4.1. Solve for tuple $(\mathbf{w}, \mathbf{x})$ at fixed U^*

As $\mathbf{x}$ is shared across the two terms in the objective in (3), we introduce a set of slack variables $\{\mathbf{z}^i\}$ for each $\mathbf{D}^i\mathbf{x}$. This gives the following sub-problem

$$\begin{aligned} \min_{\mathbf{w}, \mathbf{x}} \quad & \sum_{i=1}^{l} \mu_i \|\mathbf{z}^i\|_2 + \frac{\lambda}{2} \|U^*\mathbf{w} + \mathbf{x} - \mathbf{v}\|_2^2 \\ \text{s.t.} \quad & \mathbf{z}^i = D^i \mathbf{x}, \quad i = 1, \cdots, l. \end{aligned} \tag{4}$$

Model (4) is convex over $\{\mathbf{z}^i\}$ and $(\mathbf{w}, \mathbf{x})$, while the constraints are affine. A natural choice to solve such a problem efficiently is the Alternating Direction Method of Multipliers (ADMM) [5], assuming we can show that each resul-

tant sub-calculations can be performed cheaply. Next, we demonstrate that this is indeed the case here.

The augmented Lagrangian [28] of (4) is given by

$$\mathcal{L}(\mathbf{w},\mathbf{x},\{\mathbf{z}^i\},\{\mathbf{y}^i\}) = \sum_{i=1}^{l}\mu_i\|\mathbf{z}^i\|_2 + \frac{\lambda}{2}\|U^*\mathbf{w}+\mathbf{x}-\mathbf{v}\|_2^2 + \sum_{i=1}^{l}{\mathbf{y}^i}^T(D^i\mathbf{x}-\mathbf{z}^i) + \sum_{i=1}^{l}\frac{\rho_i}{2}\|D^i\mathbf{x}-\mathbf{z}^i\|_2^2 \tag{5}$$

Here ρ^i are predefined positive parameters, and $\mathbf{y}^i$ are the dual variables associated with the constraints. Our update scheme proceeds as follows. Given the current observation $\mathbf{v}$ and the tuple $(\mathbf{w}_k,\mathbf{x}_k,\{\mathbf{z}_k^i\},\{\mathbf{y}_k^i\})$ at k^{th} iteration, the step-by-step updating of the tuple at $(k+1)^{th}$ iteration is:

(w,x)-minimization: To minimize (5) with respect to (w,x) alone, keeping all the other parameters fixed, we have

$$\min_{\mathbf{w},\mathbf{x}} \quad \frac{\lambda}{2}\|U^*\mathbf{w}+\mathbf{x}-\mathbf{v}\|_2^2 + \sum_{i=1}^{l}{\mathbf{y}_k^i}^T D^i\mathbf{x} + \sum_{i=1}^{l}\frac{\rho_i}{2}\|D^i\mathbf{x}-\mathbf{z}_k^i\|_2^2 \tag{6}$$

(6) takes the form of a convex quadratic problem in $(\mathbf{w},\mathbf{x})$ and the closed form solution comes from the linear system, $A\begin{bmatrix}\mathbf{w} & \mathbf{x}\end{bmatrix}^T = \mathbf{b}$. Note that ${D^i}^T D^i = {D^i}^T = D^i$, and $A,\mathbf{b}$ are computed as line 2 and 3 in Algorithm 1. Solving this linear system directly can be computational expensive when n is large. However, observing the structure of A, we have the following result. All of our proofs are included in the extended version:

Observation 1. *For $\lambda > 0, {U^*}^T U^* = I_d, \rho_i > 0, \forall i \in \{1,\cdots,l\}$, we have $A \succ 0$.*

Together with the fact that A is sparse, we use a GPU solver using preconditioned conjugate gradient method [28], which reduces the running time significantly.

$\mathbf{z}^i$-minimization: Minimizing a specific $\mathbf{z}^i$ for group i, is independent of the other $\mathbf{z}^{j\neq i}$ and hence can be solved in parallel. The objective w.r.t $\mathbf{z}^i$ takes the form,

$$\min_{\mathbf{z}^i} \quad \mu_i\|\mathbf{z}^i\|_2 - {\mathbf{y}_k^i}^T\mathbf{z}^i + \frac{\rho_i}{2}\|D^i\mathbf{x}_{k+1}-\mathbf{z}^i\|_2^2 \tag{7}$$

Denoting $\mathbf{r}_k^i = \mathbf{D}^i\mathbf{x}_{k+1} + \frac{{\mathbf{y}^i}_k}{\rho_i}$, (7) has a closed form solution by the block soft thresholding formula [39] given as,

$$\mathbf{z}_{k+1}^i = \max\{\|\mathbf{r}_k^i\|_2 - \frac{\mu_i}{\rho_i}, 0\}\frac{\mathbf{r}_k^i}{\|\mathbf{r}_k^i\|_2} \tag{8}$$

$\mathbf{y}^i$-updating: We can now update $y^i, \forall i \in \{1,\cdots,l\}$ along the gradient direction by,

$${\mathbf{y}^i}_{k+1} = \mathbf{y}_k^i + \rho_i(D^i\mathbf{x}_{k+1} - \mathbf{z}_{k+1}^i) \tag{9}$$

The above analysis shows that the key update steps (summarized in Algorithm 1) within a ADMM procedure can all be performed efficiently. In our implementation, we alternatively solve for $(\mathbf{w}^*,\mathbf{x}^*,\mathbf{z}^{i*},\mathbf{y}^*)$ until the changes in $\mathbf{x}$ and the objective value reaches a desired level of tolerance. Given the convexity of each item in the tuple, we have the following convergence theorem.

Theorem 1. *For $\lambda > 0, \mu_i > 0, \rho_i > 0, \forall i \in \{1,\cdots,l\}$, the sequence $\{(\mathbf{w}_k,\mathbf{x}_k,\{\mathbf{z}_k^i\},\{\mathbf{y}^i\})\}$ generated by Alg. 1 from any initial point $(\mathbf{w}_0,\mathbf{x}_0,\{\mathbf{z}_0^i\},\{\mathbf{y}_0^i\})$ converges to $(\mathbf{w}^*,\mathbf{x}^*,\{\mathbf{z}^{i*}\},\{\mathbf{y}^{i*}\})$, which minimizes (5) at fixed U^*.*

Algorithm 1 ADMM for solving $(\mathbf{w}^*,\mathbf{x}^*)$

In: Subspace matrix: U^*, observation: $\mathbf{v}$, initial: $\mathbf{x}_0, \mathbf{z}_0^i, \mathbf{y}_0^i$, group operator: D^i, hyper-parameters: λ, μ, ρ
Out: Subspace coefficient: $\mathbf{w}^*$, structured outliers: $\mathbf{x}^*$
Procedure:
1: **for** $k = 0 \to K$ **do**
2: $A \leftarrow \begin{bmatrix}\lambda I_d & \lambda {U^*}^T \\ \lambda U^* & \lambda I_n + \sum_{i=1}^{l}\rho_i D^i\end{bmatrix}$;
3: $\mathbf{b} \leftarrow \begin{bmatrix}\lambda {U^*}^T\mathbf{v} \\ \lambda\mathbf{v} - \sum_{i=1}^{l}D^i\mathbf{y}_k^i + \sum_{i=1}^{l}\rho_i D^i\mathbf{z}_k^i\end{bmatrix}$
4: $(\mathbf{w}_{k+1},\mathbf{x}_{k+1}) \leftarrow \min_{\mathbf{w},\mathbf{x}}\|(A[\mathbf{w}\quad\mathbf{x}]^T - \mathbf{b})\|^2$ using GPU solver
5: $\mathbf{r}_k^i \leftarrow \mathbf{D}^i\mathbf{x}_{k+1} + \frac{{\mathbf{y}^i}_k}{\rho_i}$
6: $\mathbf{z}_{k+1}^i \leftarrow \max\{\|\mathbf{r}_k^i\|_2 - \frac{\mu_i}{\rho_i}, 0\}\frac{\mathbf{r}_k^i}{\|\mathbf{r}_k^i\|_2}$
7: ${\mathbf{y}^i}_{k+1} \leftarrow \mathbf{y}_k^i + \rho_i(D^i\mathbf{x}_{k+1} - \mathbf{z}_{k+1}^i)$
8: Stop if tolerance conditions satisfied.
9: **end for**

4.2. Update of U with estimated $(\mathbf{w}^*,\mathbf{x}^*)$

The key idea to update U is to refine it from the estimation $(\mathbf{w}^*,\mathbf{x}^*)$ derived from the current observation $\mathbf{v}$ on the Grassmannian. Given the estimated tuple $(\mathbf{w}^*,\mathbf{x}^*)$, the derivative of $\mathcal{L}(.)$ in (5) with respect to the components of U and the gradient are given by

$$\frac{\partial\mathcal{L}}{\partial U} = \lambda(U\mathbf{w}^* + \mathbf{x}^* - \mathbf{v}){\mathbf{w}^*}^T = \mathbf{s}{\mathbf{w}^*}^T \tag{10}$$

where $\mathbf{s} = \lambda(U\mathbf{w}^* + \mathbf{x}^* - \mathbf{v})$ denotes the residual vector. Using identity (2.70) in [2], the gradient on the Grassmannian can be computed by

$$\nabla\mathcal{L} = (I - UU^T)\frac{\partial\mathcal{L}}{\partial U} = (I-UU^T)\mathbf{s}{\mathbf{w}^*}^T = \mathbf{s}{\mathbf{w}^*}^T \tag{11}$$

(11) is valid because the residual vector $\mathbf{s}$ is orthogonal to all of the columns of U. It is obvious that $\nabla\mathcal{L}$ is a rank one matrix, since $\mathbf{s}$ and $\mathbf{w}^*$ are both vectors. Hence, we can compute the compact SVD of $\nabla\mathcal{L}$ by $\nabla\mathcal{L} = \mathbf{p}\sigma\mathbf{q}$, where $\mathbf{p} = \frac{\mathbf{s}}{\|\mathbf{s}\|}$, $\sigma = \|\mathbf{s}\|\|\mathbf{w}^*\|$ and $\mathbf{q} = \frac{\mathbf{w}^*}{\|\mathbf{w}^*\|}$. Following [2, 15], we update U with a gradient stepsize η in the direction $-\nabla\mathcal{L}$ as

$$U(\eta) = U + (\cos(\sigma\eta) - 1)U\mathbf{q}\mathbf{q}^T - \sin(\sigma\eta)\mathbf{p}\mathbf{q}^T \tag{12}$$

where η is the stepsize to update the subspace U on the Grassmann manifold. We incorporate an adaptive stepsize η using the updating scheme by [20] but in the experiments, a constant stepsize works well also. To show the validity of (12), we give the following lemma,

Lemma 1. *The subspace updating procedure (12) preserves the column-wise orthogonality of U.*

Notice that (12) is related to a stochastic gradient updating procedure, where at each iteration, we draw an example in a sequential manner, instead of random sampling. We compute the gradient from each example, and use this gradient to improve the subspace. The optimal subspace is not computed fully, and is instead updated by analyzing successive observations. Additional details on (12) are given in the extended version. At this point, we are ready to summarize our optimization pipeline in Algorithm 2.

Algorithm 2 Main Procedure of GOSUS

In: Observation: V, subspace initialization: U_0, hyperparameters: λ, μ, ρ
Out: Approximated signal: B, structured outliers: X
Procedure:

1: **for** $t = 1 \rightarrow T$ **do**
2: Solve $(\mathbf{w}^*, \mathbf{x}^*, \{\mathbf{z}^{i*}\}, \{\mathbf{y}^{i*}\})$ by Algorithm 1;
3: (Optional) Update stepsize η_t ;
4: Update U_t by (12);
5: **end for**

5. Applications

We apply GOSUS to the problem of foreground/background separation and multiple face tracking/identity management. Our implementation and experiments are publicly available.

5.1. Background Subtraction

Datasets. We used two benchmark datasets: Perception Test Images Sequences [22] and Wallflower Test Images Sequences [31], which are heavily used in recent work [26, 29, 15, 36]. The data includes 12 video sequences, with a variety of characteristics, such as changing foreground with static (Bootstrap, Shopping Mall, Hall) and dynamic (Fountain, Escalator, Waving Trees, Water Surface, Curtain, Campus) backgrounds as well as illumination changes (Lobby, Time of Day, Light switch).

Experiments setup. GOSUS is compared to three different models: **(i)** Batch model: (RPCA) Robust PCA using Inexact Augmented Lagrange Multiplier Method [24] **(ii)** Batch model: (RPMF) Robust Probabilistic Matrix Factorization [36], **(iii)** Online model: (GRASTA) Grassmannian Robust Adaptive Subspace Tracking [15]. For these baseline methods, we use code from the corresponding authors' websites. For RPCA, the maximum number of iterations was set to 1000 and the regularization parameter was $\frac{1}{\gamma}$ (γ is the number of pixels in the image frame). The regularization parameters (one for each of the two factorizing matrices) in RPMF were set to 1. To obtain best possible results from GRASTA, sub-sampling was turned off and the code was initialized with the suggested default settings. In GOSUS, for each color frame, we extract a vector $\mathbf{v}$ with size n (i.e, # of pixels times 3 for the RGB channels). The ADMM hyperparameters used were $\rho^i = 0.3/\text{mean}(\mathbf{v}), \forall i = 1, \cdots, l$ and stepsize η was 0.01. λ was set using cross–validation and all μ_is were set to 1. An initial estimate of the background subspace was set as a random orthonormal matrix $n \times d$ (where $d = 5$, n is equal to three times # of pixels in each frame). The tolerance level for all methods was set at 10^{-6}. Note that RPCA and RPMF see all the data at once which gives them an inherent advantage over GRASTA. Receiver Operating Characteristic (ROC) curves, and the corresponding area under curve (AUC) values are used as performance evaluation measures.

Video Datasets	Models RPCA[24]	RPMF[36]	GRASTA[15]	**GOSUS**
Fountain	0.94	0.94	0.69	**0.99**
Escalator	0.91	0.90	0.90	**0.96**
WavingTrees	0.74	0.84	0.87	**0.98**
Campus	0.90	0.86	0.77	**0.98**
Bootstrap	0.87	0.91	0.87	**0.93**
WaterSurface	0.73	0.84	0.87	**0.97**
Hall	0.82	0.90	0.76	**0.93**
Time of Day	0.80	0.85	0.84	**0.89**
LightSwitch	0.87	**0.92**	0.62	0.88
Curtain	0.87	0.90	0.88	**0.96**
Lobby	0.89	0.94	0.70	**0.95**
ShoppingMall	0.92	0.93	0.90	**0.94**

Table 1: Area under ROC curves for RPCA, RPMF, GRASTA, GOSUS.

Group Construction. Together with a 3×3 grid group structure (patches) and hierarchical tree group structure [19], we also use a coarse-to-fine superpixel group construction. Pixels belonging to each superpixel form a group which can overlap with others. We employ the SLIC superpixel algorithm, with region sizes $\{80, 40, 20, 10\}$ in order to generate coarse-to-fine groups [1]. The group construction captures the boundary information of objects and our evaluations show this setting works well.

Quantitative Evaluations. Figure 1 summarizes the ROC plots for 6 videos, representative examples from the three different data categories that constitute our data. Table 1 presents the AUC values for all 12 videos. The results indicate that GOSUS performs better than all baseline methods (except on the 'Light Switch' video where RPMF was the best). In particular, from Table 1 we see that GOSUS competes very favorably with GRASTA, both being online methods. This is particularly clear in data with dynamic background (Fountain, Campus) and illumination changes (Light Switch, Lobby). Also note that RPCA and RPMF are batch models, and GOSUS attains better performance than either in almost all categories, which supports the intuition that imposing structure (spatial homogeneity) on the outliers enables it to improve estimating the subspace.

Qualitative Evaluations. Figure 2 shows the effectiveness

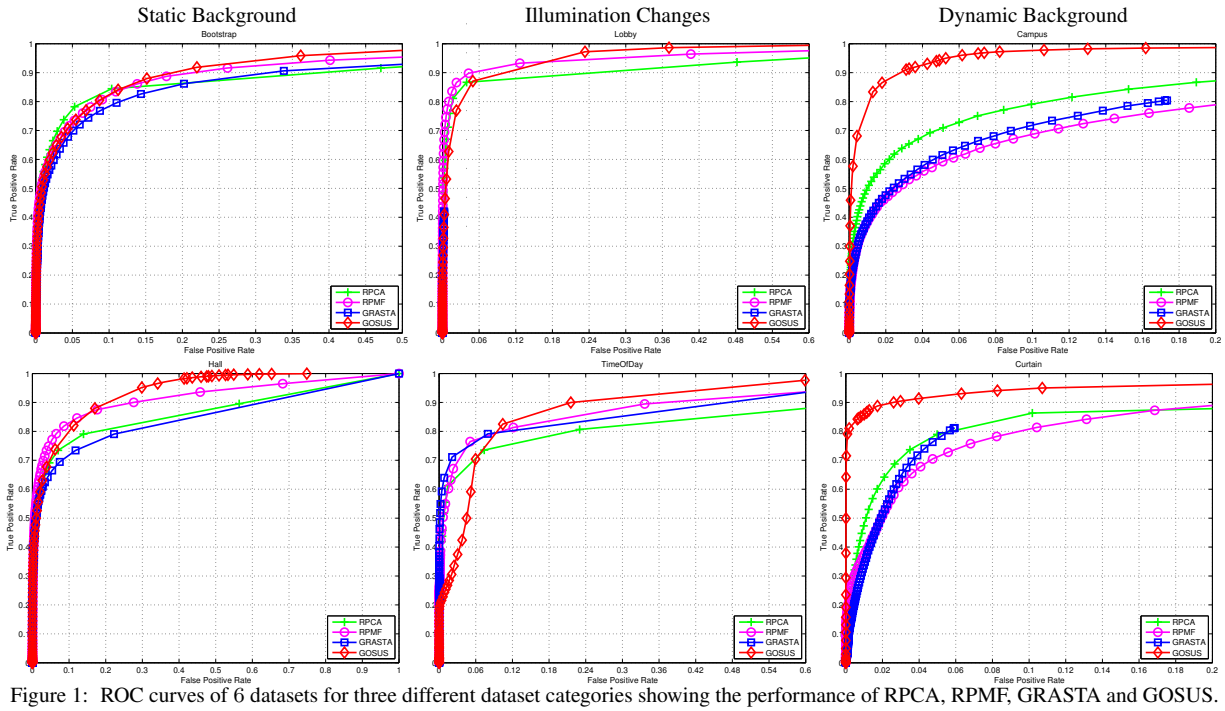

Figure 1: ROC curves of 6 datasets for three different dataset categories showing the performance of RPCA, RPMF, GRASTA and GOSUS.

of GOSUS in adapting to intermittent object motion in the background. GOSUS starts with a random subspace and finds the correct background after 200 frames. At frame t_0+645, a person comes in, sits for a while, and leaves on frame $t_0 + 882$. GOSUS successfully learn the new background (notice the pose of the red chair) as early as frame $t_0 + 930$.

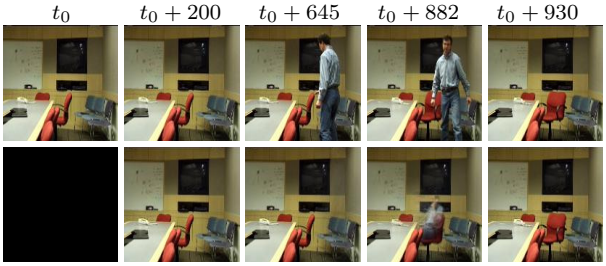

Figure 2: Effectiveness on adapting to intermittent object motion in the background. The first row are the original frames, and the second row are the background learned by GOSUS.

Figure 3 shows example detections for four different videos (one frame for each) of our algorithm and several baselines. The first row corresponds to an example with static background, and GOSUS performs comparably with others. The last three videos have dynamic background, where the water surface is moving, trees are swaying, etc. Observe that outputs of GOSUS contain very few isolated foreground regions, unlike GRASTA and the other batch models RPCA and RPMF, which do not regularize the secondary term at all. Further, the foreground object by itself is better segmented (very few pixels missing along the boundaries) in GOSUS. This shows that the structured sparsity used in GOSUS, is not only acting as a noise removal filter (on salt-and-pepper like foreground detections) but also improves the estimation of the perturbed (dynamic/moving) subspace. Further note that GOSUS outperforms both batch models (RPCA and RPMF), since the latter do not use any form of spatial contiguity. Overall, both Table 1 and Figure 3 indicate that GOSUS improves background subtraction in various categories, and offers substantial improvements when the background is dynamic.

We also compare GOSUS with sparse coding based methods. As shown in Figure 4, our method is competitive with [26], except there are some grid artifacts from [26] due to their group construction. However, our algorithm achieves $1 \sim 2$ frames per second given the original image size (no resizing). This is significantly faster than the bi-level process used in [26], and several orders of magnitude faster than speed reported in [29], a method devoted to optimizing structured sparsity norm.

Figure 4: Comparison with [26] using overlapping groups.

5.2. Multiple Face Tracking/Identity Management

Our second application is to track multiple faces (keeping track of the identities) in real world videos, e.g., TV shows and movies. This problem is extremely challenging due to the dramatic variation in the appearance of each person's face, and the dynamics of characters coming in and

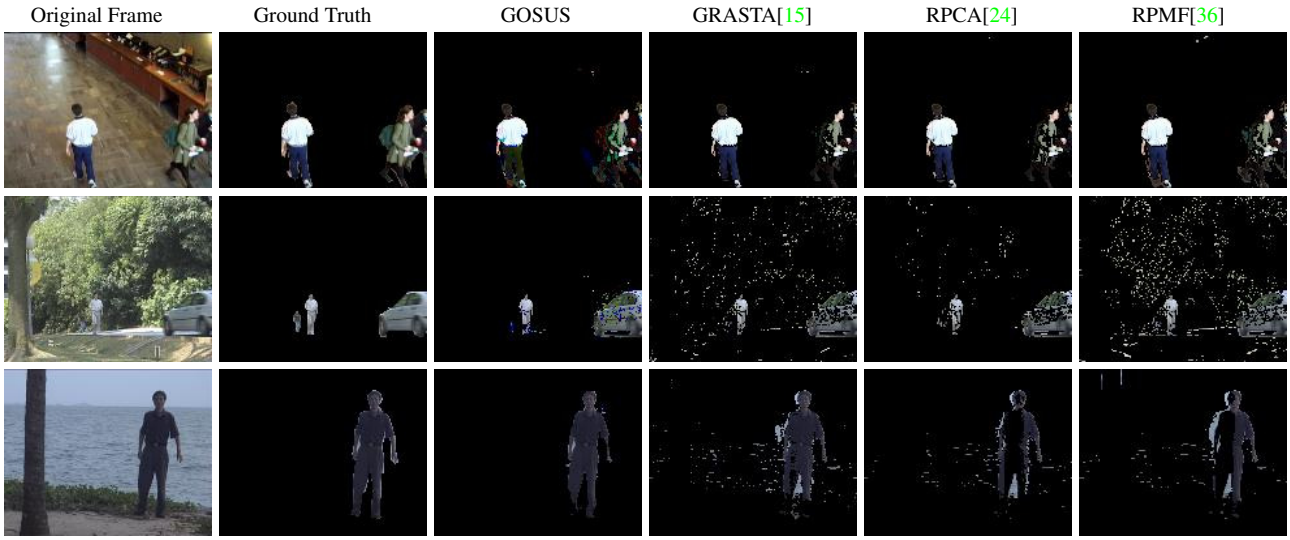

Figure 3: Example results on Bootstrap, Campus, and Water Surface comparing GOSUS to ground truth followed by GRASTA, RPCA and RPMF.

out. Existing work has achieved the state of art by utilizing all visual frames, audio, aligned subtitle and script texts [12, 30]. We aim to tackle this problem using *only* the visual data, and in an efficient manner.

We first run Viola-Jones detector [35] on all image frames. For robustness to pose/expression variation, lighting, and partial occlusion, we use a parts-based descriptor extracted around detected facial features [12, 30]. We detect 13 facial feature points (the left and right corners, center of each eye and mouth, the two nostrils, tip of the nose, center of the eyes) and simply extract a pixel-wise descriptor of the circular region around each feature point (which we transform on to a canonical face). This gives us a 1937 dimensional feature vector $\mathbf{v}$ for each face. The structured sparsity prior refers to each circular region as a group. This setting can capture the occlusion created by glasses/shadows as well as self-occlusions due to pose variations.

The tracking and identity management procedure is related to face recognition approaches reported in [33, 19]. We consider U as a face subspace, with each column representing an 'eigenface'. The observed face vector is described by a combination of eigenfaces using $\mathbf{w}$ and structured outliers $\mathbf{x}$, created by occlusion/disguise. $\mathbf{w}$ acts as a signature for each face. False positives from the face detector are rejected by thresholding the norm of $\mathbf{x}$. We maintain a window (size 400) for tracked faces. The label for each face (i.e., identity) comes from a majority nearest neighbor votes from this window, along with temporal consistency. When a new face is found, we add a new label/identity to our signature window.

We demonstrate the effectiveness of GOSUS on several real world videos from the TV show: 'The Big Bang Theory'. Sample results are shown in Figure 5. Faces marked with the same number are from the same track. Firstly observe that Amy in frame 151 and frame 1009, is tracked correctly even with significant changes in camera shot. The person marked 7 (Penny) is also correctly tracked over a long time (frame 1297 through 2012 to 3693). However, different tracks for the same person may be introduced if the person (Rajesh/Sheldon marked as 3/4) disappears in the video for a long time or has dramatic facial expressions.

Though our preliminary application on multiple face tracking shows promising results for real videos, the current pipeline is limited (in terms of efficiency) to the output from the face detector. On these videos (720×1280), it takes about 2 seconds to detect all possible faces (for each frame), whereas GOSUS on its own can process all 6000 frames with all detected faces in $\sim$ 20 seconds. Also note that the face detector can only detect frontal faces (the face of the male in frame 151 is missing), and can introduce a sizeable number of false positives for real world videos. Improvements to these modules will seamlessly yield improvements in the empirical performance of GOSUS.

6. Conclusion

The main contribution of this paper is an intuitive yet expressive model, GOSUS, which exploits a meaningful structured sparsity term to significantly improve the accuracy of online subspace updates. We discuss the modeling and optimization aspects in detail. Our solution is based on ADMM, where most key steps in the update procedure reduce to simple matrix operations yielding real-time performance for several interesting problems in video analysis.

Acknowledgments: We thank Laura Balzano, Maxwell Collins, and anonymous reviewers for helpful comments. This research is funded via grants NSF RI 1116584, NSF CGV 1219016, NSF Award 0916687, and NSF EA 1029679. Partial support was provided by UW-ICTR and UW Graduate School/WARF grant.

Figure 5: Examples of multiple face tracking in the Big Bang Theory. Faces marked with the same number are from the same track. Frame number is shown on the left top corner. Complete video results are provided on the project website.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *PAMI*, 34(11):2274–2282, 2012.

[2] T. Arias, A. Edelman, and S. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20:303–353, 1998.

[3] S. Arora, R. Ge, R. Kannan, and A. Moitra. Computing a nonnegative matrix factorization – provably. In *STOC*, 2012.

[4] L. Balzano, R. Nowak, and B. Recht. Online identification and tracking of subspaces from highly incomplete information. In *Proceedings of the Allerton Conference on Communication*, 2010.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[6] S. Bucak and B. Gunsel. Incremental subspace learning via nonnegative matrix factorization. *Pattern Recog.*, 42(5):788–797, 2009.

[7] D. Cai, X. He, and J. Han. Spectral regression for efficient regularized subspace learning. In *ICCV*, 2007.

[8] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *J. ACM*, 58(3):11, 2011.

[9] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

[10] X. Chen, Q. Lin, S. Kim, J. G. Carbonell, and E. P. Xing. Smoothing proximal gradient method for general structured sparse learning. In *UAI*, 2011.

[11] F. De La Torre and M. Black. A framework for robust subspace learning. *IJCV*, 54(1):117–142, 2003.

[12] M. Everingham, J. Sivic, and A. Zisserman. "Hello! My name is Buffy" – automatic naming of characters in TV video. In *BMVC*, 2006.

[13] P. Favaro, R. Vidal, and A. Ravichandran. A closed form solution to robust subspace estimation and clustering. In *CVPR*, 2011.

[14] J. Hamm and D. D. Lee. Grassmann discriminant analysis: a unifying view on subspace-based learning. In *ICML*, 2008.

[15] J. He, L. Balzano, and A. Szlam. Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In *CVPR*, 2012.

[16] J. Huang, X. Huang, and D. N. Metaxas. Learning with dynamic group sparsity. In *ICCV*, 2009.

[17] J. Huang and T. Zhang. The benefit of group sparsity. *Annals of Statistics*, 38:1978–2004, 2010.

[18] A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. *Neural Networks*, 13(4):411–430, 2000.

[19] K. Jia, T.-H. Chan, and Y. Ma. Robust and practical face recognition via structured sparsity. In *ECCV*, 2012.

[20] S. Klein, J. P. W. Pluim, M. Staring, and M. A. Viergever. Adaptive stochastic gradient descent optimisation for image registration. *IJCV*, 81(3):227–239, 2009.

[21] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, 2011.

[22] L. Li, W. Huang, I. Y. H. Gu, and Q. Tian. Statistical modeling of complex backgrounds for foreground object detection. *TIP*, 13(11):1459–1472, 2004.

[23] S. Z. Li, X. Lu, X. Hou, X. Peng, and Q. Cheng. Learning multiview face subspaces and facial pose estimation using independent component analysis. *TIP*, 14(6):705–712, 2005.

[24] Z. Lin, M. Chen, L. Wu, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv*, 1009.5055, 2010.

[25] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *JMLR*, 11:19–60, 2010.

[26] J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Convex and network flow optimization for structured sparsity. *JMLR*, 12:2681–2720, 2011.

[27] L. Mukherjee, V. Singh, J. Xu, and M. D. Collins. Analyzing the subspace structure of related images: Concurrent segmentation of image sets. In *ECCV*, 2012.

[28] J. Nocedal and S. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering, 2006.

[29] Z. Qin and D. Goldfarb. Structured sparsity via alternating directions methods. *JMLR*, 13:1373–1406, 2012.

[30] J. Sivic, M. Everingham, and A. Zisserman. "Who are you?" – learning person specific classifiers from video. 2009.

[31] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *ICCV*, 1999.

[32] P. Turaga, A. Veeraraghavan, and R. Chellappa. Statistical analysis on stiefel and grassmann manifolds with applications in computer vision. In *CVPR*, 2008.

[33] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *CVPR*, pages 586–591, 1991.

[34] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (GPCA). *PAMI*, 27(12):1945–1959, 2005.

[35] P. A. Viola and M. J. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.

[36] N. Wang, T. Yao, J. Wang, and D.-Y. Yeung. A probabilistic approach to robust matrix factorization. In *ECCV*, 2012.

[37] T. Wang, A. Backhouse, and I. Gu. Online subspace learning on grassmann manifold for moving object tracking in video. In *Int. Conf. Acoustics, Speech, and Signal Processing*, 2008.

[38] K. Yu, Y. Lin, and J. Lafferty. Learning image representations from the pixel level via hierarchical sparse coding. In *CVPR*, 2011.

[39] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.