Supervised Descent Method and its Applications to Face Alignment

Xuehan Xiong **Fernando De la Torre**
The Robotics Institute, Carnegie Mellon University, Pittsburgh PA, 15213
xxiong@andrew.cmu.edu ftorre@cs.cmu.edu

Abstract

Many computer vision problems (e.g., camera calibration, image alignment, structure from motion) are solved through a nonlinear optimization method. It is generally accepted that 2^{nd} order descent methods are the most robust, fast and reliable approaches for nonlinear optimization of a general smooth function. However, in the context of computer vision, 2^{nd} order descent methods have two main drawbacks: (1) The function might not be analytically differentiable and numerical approximations are impractical. (2) The Hessian might be large and not positive definite.

To address these issues, this paper proposes a Supervised Descent Method (SDM) for minimizing a Non-linear Least Squares (NLS) function. During training, the SDM learns a sequence of descent directions that minimizes the mean of NLS functions sampled at different points. In testing, SDM minimizes the NLS objective using the learned descent directions without computing the Jacobian nor the Hessian. We illustrate the benefits of our approach in synthetic and real examples, and show how SDM achieves state-of-the-art performance in the problem of facial feature detection. The code is available at www.humansensing.cs.cmu.edu/intraface.

1. Introduction

Mathematical optimization has a fundamental impact in solving many problems in computer vision. This fact is apparent by having a quick look into any major conference in computer vision, where a significant number of papers use optimization techniques. Many important problems in computer vision such as structure from motion, image alignment, optical flow, or camera calibration can be posed as solving a nonlinear optimization problem. There are a large number of different approaches to solve these continuous nonlinear optimization problems based on first and second order methods, such as gradient descent [1] for dimensionality reduction, Gauss-Newton for image alignment [22, 5, 14] or Levenberg-Marquardt for structure from motion [8].

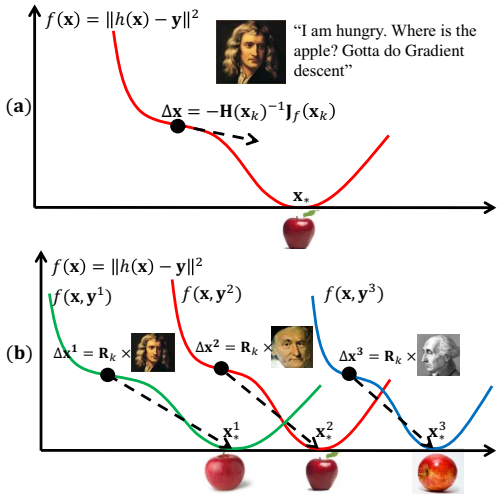


Figure 1: a) Using Newton's method to minimize $f(\mathbf{x})$. b) SDM learns from training data a set of generic descent directions $\{\mathbf{R}_k\}$. Each parameter update ($\Delta\mathbf{x}^i$) is the product of $\mathbf{R}_k$ and an image-specific component ($\mathbf{y}^i$), illustrated by the 3 great Mathematicians. Observe that no Jacobian or Hessian approximation is needed at test time. We dedicate this figure to I. Newton, C. F. Gauss, and J. L. Lagrange for their everlasting impact on today's sciences.

Despite its many centuries of history, the Newton's method (and its variants) is regarded as a major optimization tool for smooth functions when second derivatives are available. Newton's method makes the assumption that a smooth function $f(\mathbf{x})$ can be well approximated by a quadratic function in a neighborhood of the minimum. If the Hessian is positive definite, the minimum can be found by solving a system of linear equations. Given an initial estimate $\mathbf{x}_0 \in \Re^{p\times 1}$, Newton's method creates a sequence of updates as

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{H}^{-1}(\mathbf{x}_k)\mathbf{J}_f(\mathbf{x}_k), \quad (1)$$

where $\mathbf{H}(\mathbf{x}_k) \in \Re^{p\times p}$ and $\mathbf{J}_f(\mathbf{x}_k) \in \Re^{p\times 1}$ are the Hessian matrix and Jacobian matrix evaluated at $\mathbf{x}_k$. Newton-type methods have two main advantages over competitors. First, when it converges, the convergence rate is quadratic. Second, it is guaranteed to converge provided that the initial estimate is sufficiently close to the minimum.

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.75

However, when applying Newton's method to computer vision problems, three main problems arise: (1) The Hessian is positive definite at the local minimum, but it might not be positive definite elsewhere; therefore, the Newton steps might not be taken in the descent direction. (2) Newton's method requires the function to be twice differentiable. This is a strong requirement in many computer vision applications. For instance, consider the case of image alignment using SIFT [21] features, where the SIFT can be seen as a non-differentiable image operator. In these cases, we can estimate the gradient or the Hessian numerically, but this is typically computationally expensive. (3) The dimension of the Hessian matrix can be large; inverting the Hessian requires $O(p^3)$ operations and $O(p^2)$ in space, where p is the dimension of the parameter to estimate. Although explicit inversion of the Hessian is not needed using Quasi-Netwon methods such as L-BFGS [9], it can still be computationally expensive to use these methods in computer vision problems. In order to address previous limitations, this paper proposes a Supervised Descent Method (SDM) that learns the descent directions in a supervised manner.

Fig. 1 illustrates the main idea of our method. The top image shows the application of Newton's method to a Non-linear Least Squares (NLS) problem, where $f(\mathbf{x})$ is a non-linear function and $\mathbf{y}$ is a known vector. In this case, $f(\mathbf{x})$ is a non-linear function of image features (*e.g.*, SIFT) and $\mathbf{y}$ is a known vector (*i.e.*, template). $\mathbf{x}$ represents the vector of motion parameters (*i.e.*, rotation, scale, non-rigid motion). The traditional Newton update has to compute the Hessian and the Jacobian. Fig. 1b illustrates the main idea behind SDM. The training data consists of a set of functions $\{f(\mathbf{x}, \mathbf{y}^i)\}$ sampled at different locations $\mathbf{y}^i$ (*i.e.*, different people) where the minima $\{\mathbf{x}^i_*\}$ are known. Using this training data, SDM learns a series of parameter updates, which incrementally, minimizes the mean of all NLS functions in training. In the case of NLS, such updates can be decomposed into two parts: a sample specific component (*e.g.*, $\mathbf{y}^i$) and a generic descent directions $\mathbf{R}_k$. SDM learns average descent directions $\mathbf{R}_k$ during training. In testing, given an unseen $\mathbf{y}$, an update is generated by projecting $\mathbf{y}$-specific components onto the learned generic directions $\mathbf{R}_k$.

We illustrate the benefits of SDM on analytic functions, and in the problem of facial feature detection and tracking. We show how SDM improves state-of-the-art performance for facial feature detection in two "face in the wild" databases [26, 4] and demonstrate extremely good performance tracking faces in the YouTube celebrity database [20].

2. Previous work

This section reviews previous work on face alignment.

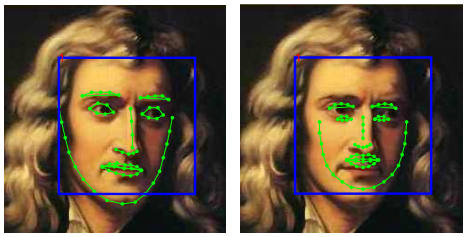
Parameterized Appearance Models (PAMs), such as Active Appearance Models [11, 14, 2], Morphable Models [6, 19], Eigentracking [5], and template tracking [22, 30] build an object appearance and shape representation by computing Principal Component Analysis (PCA) on a set of manually labeled data. Fig. 2a illustrates an image labeled with p landmarks ($p = 66$ in this case). After the images are aligned with Procrustes, the shape model is learned by computing PCA on the registered shapes. A linear combination of k_s shape basis, $\mathbf{U}^s \in \Re^{2p \times k_s}$ can reconstruct (approximately) any aligned shape in the training set. Similarly, an appearance model, $\mathbf{U}^a \in \Re^{m \times k_a}$, is built by performing PCA on the texture. Alignment is achieved by finding the motion parameter $\mathbf{p}$ and appearance coefficients $\mathbf{c}^a$ that best aligns the image w.r.t. the subspace $\mathbf{U}^a$, *i.e.*,

$$\min_{\mathbf{c^a},\mathbf{p}} ||\mathbf{d}(\mathbf{f}(\mathbf{x},\mathbf{p})) - \mathbf{U}^a\mathbf{c}^a||_2^2, \tag{2}$$

$\mathbf{x} = [x_1, y_1, ...x_l, y_l]^\top$ is the vector containing the coordinates of the pixels to detect/track. $\mathbf{f}(\mathbf{x}, \mathbf{p})$ represents a geometric transformation; the value of $\mathbf{f}(\mathbf{x}, \mathbf{p})$ is a vector denoted by $[u_1, v_1, ..., u_l, v_l]^\top$. $\mathbf{d}(\mathbf{f}(\mathbf{x}, \mathbf{p}))$ is the appearance vector of which the i^{th} entry is the intensity of image $\mathbf{d}$ at pixel (u_i, v_i). For affine and non-rigid transformations, (u_i, v_i) relates to (x_i, y_i) by $\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ a_4 & a_5 \end{bmatrix} \begin{bmatrix} x^s_i \\ y^s_i \end{bmatrix} + \begin{bmatrix} a_3 \\ a_6 \end{bmatrix}$. Here $[x^s_1, y^s_1, ...x^s_l, y^s_l]^\top = \overline{\mathbf{x}} + \mathbf{U}^s\mathbf{c}^s$, where $\overline{\mathbf{x}}$ is the mean shape face. $\mathbf{a}, \mathbf{c}^s$ are affine and non-rigid motion parameters respectively and $\mathbf{p} = [\mathbf{a}; \mathbf{c}^s]$.

Given an image $\mathbf{d}$, PAMs alignment algorithms optimize Eq. 2. Due to the high dimensionality of the motion space, a standard approach to efficiently search over the parameter space is to use the Gauss-Newton method [5, 2, 11, 14] by doing a Taylor series expansion to approximate $\mathbf{d}(\mathbf{f}(\mathbf{x}, \mathbf{p} + \Delta\mathbf{p})) \approx \mathbf{d}(\mathbf{f}(\mathbf{x}, \mathbf{p})) + \mathbf{J_d}(\mathbf{p})\Delta\mathbf{p}$, where $\mathbf{J_d}(\mathbf{p}) = \frac{\partial \mathbf{d}(\mathbf{f}(\mathbf{x},\mathbf{p}))}{\partial \mathbf{p}}$ is the Jacobian of the image $\mathbf{d}$ w.r.t. to the motion parameter $\mathbf{p}$ [22].

Discriminative approaches learn a mapping from image features to motion parameters or landmarks. Cootes *et al.* [11] proposed to fit AAMs by learning a linear regression between the increment of motion parameters $\Delta\mathbf{p}$ and the appearance differences $\Delta\mathbf{d}$. The linear regressor is a numerical approximation of the Jacobian [11]. Following this idea, several discriminative methods that learn a mapping from $\mathbf{d}$ to $\Delta\mathbf{p}$ have been proposed. Gradient Boosting, first introduced by Friedman [16], has become one of the most popular regressors in face alignment because of its efficiency and the ability to model nonlinearities. Saragih and Göcke [27] and Tresadern *et al.* [29] showed that using boosted regression for AAM discriminative fitting significantly improved over the original linear formulation. Dollár *et al.* [15] incorporated "pose indexed features" to the boosting framework, where not only



(a) $\mathbf{x}_*$ (b) $\mathbf{x}_0$

Figure 2: a) Manually labeled image with 66 landmarks. Blue outline indicates face detector. b) Mean landmarks, $\mathbf{x}_0$, initialized using the face detector.

a new weak regressor is learned at each iteration but also the features are re-computed at the latest estimate of the landmark location. Beyond the gradient boosting, Rivera and Martinez [24] explored kernel regression to map from image features directly to landmark location achieving surprising results for low-resolution images. Recently, Cootes *et al.* [12] investigated Random Forest regressors in the context of face alignment. At the same time, Sánchez *et al.* [25] proposed to learn a regression model in the continuous domain to efficiently and uniformly sample the motion space. In the context of tracking, Zimmermann *et al.* [32] learned a set of independent linear predictor for different local motion and then a subset of them is chosen during tracking.

Part-based deformable models perform alignment by maximizing the posterior likelihood of part locations given an image. The objective function is composed of the local likelihood of each part times a global shape prior. Different methods typically vary the optimization methods or the shape prior. Constrained Local Models (CLM) [13] model this prior similarly as AAMs assuming all faces lie in a linear subspace expanded by PCA bases. Saragih *et al.* [28] proposed a non-parametric representation to model the posterior likelihood and the resulting optimization method is reminiscent of mean-shift. In [4], the shape prior was modeled non-parametrically from training data. Recently, Saragih [26] derived a sample specific prior to constrain the output space that significantly improves over the original PCA prior. Instead of using a global model, Huang et al. [18] proposed to build separate Gaussian models for each part (*e.g.*, mouth, eyes) to preserve more detailed local shape deformations. Zhu and Ramanan [31] assumed that the face shape is a tree structure (for fast inference), and used a part-based model for face detection, pose estimation, and facial feature detection.

3. Supervised Descent Method (SDM)

This section describes the SDM in the context of face alignment, and unifies discriminative methods with PAMs.

3.1. Derivation of SDM

Given an image $\mathbf{d} \in \Re^{m\times 1}$ of m pixels, $\mathbf{d}(\mathbf{x}) \in \Re^{p\times 1}$ indexes p landmarks in the image. $\mathbf{h}$ is a non-linear feature extraction function (*e.g.*, SIFT) and $\mathbf{h}(\mathbf{d}(\mathbf{x})) \in \Re^{128p\times 1}$ in the case of extracting SIFT features. During training, we will assume that the correct p landmarks (in our case 66) are known, and we will refer to them as $\mathbf{x}_*$ (see Fig. 2a). Also, to reproduce the testing scenario, we ran the face detector on the training images to provide an initial configuration of the landmarks ($\mathbf{x}_0$), which corresponds to an average shape (see Fig. 2b). In this setting, face alignment can be framed as minimizing the following function over $\Delta\mathbf{x}$

$$f(\mathbf{x}_0 + \Delta\mathbf{x}) = \|\mathbf{h}(\mathbf{d}(\mathbf{x}_0 + \Delta\mathbf{x})) - \boldsymbol{\phi}_*\|_2^2, \qquad (3)$$

where $\boldsymbol{\phi}_* = \mathbf{h}(\mathbf{d}(\mathbf{x}_*))$ represents the SIFT values in the manually labeled landmarks. In the training images, $\boldsymbol{\phi}_*$ and $\Delta\mathbf{x}$ are known.

Eq. 3 has several fundamental differences with previous work on PAMs in Eq. 2. First, in Eq. 3 we do not learn any model of shape or appearance beforehand from training data. We align the image w.r.t. a template $\boldsymbol{\phi}_*$. For the shape, our model will be a non-parametric one, and we will optimize the landmark locations $\mathbf{x} \in \Re^{2p\times 1}$ directly. Recall that in traditional PAMs, the non-rigid motion is modeled as a linear combination of shape bases learned by computing PCA on a training set. Our non-parametric shape model is able to generalize better to untrained situations (*e.g.*, asymmetric facial gestures). Second, we use SIFT features extracted from patches around the landmarks to achieve a robust representation against illumination. Observe that the SIFT operator is not differentiable and minimizing Eq. 3 using first or second order methods requires numerical approximations (*e.g.*, finite differences) of the Jacobian and the Hessian. However, numerical approximations are very computationally expensive. The goal of SDM is to learn a series of descent directions and re-scaling factors (done by the Hessian in the case of Newton's method) such that it produces a sequence of updates ($\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}_k$) starting from $\mathbf{x}_0$ that converges to $\mathbf{x}_*$ in the training data.

Now, only for derivation purposes, we will assume that $\mathbf{h}$ is twice differentiable. Such assumption will be dropped at a later part of the section. Similar to Newton's method, we apply a second order Taylor expansion to Eq. 3 as,

$$f(\mathbf{x}_0 + \Delta\mathbf{x}) \approx f(\mathbf{x}_0) + \mathbf{J}_f(\mathbf{x}_0)^\top \Delta\mathbf{x} + \frac{1}{2}\Delta\mathbf{x}^\top \mathbf{H}(\mathbf{x}_0)\Delta\mathbf{x}, \quad (4)$$

where $\mathbf{J}_f(\mathbf{x}_0)$ and $\mathbf{H}(\mathbf{x}_0)$ are the Jacobian and Hessian matrices of f evaluated at $\mathbf{x}_0$. In the following, we will omit $\mathbf{x}_0$ to simplify the notation. Differentiating (4) with respect to $\Delta\mathbf{x}$ and setting it to zero gives us the first update for $\mathbf{x}$,

$$\Delta\mathbf{x}_1 = -\mathbf{H}^{-1}\mathbf{J}_f = -2\mathbf{H}^{-1}\mathbf{J}_\mathbf{h}^\top(\boldsymbol{\phi}_0 - \boldsymbol{\phi}_*), \qquad (5)$$

where we made use of the chain rule to show that $\mathbf{J}_f = 2\mathbf{J}_\mathbf{h}^\top(\phi_0 - \phi_*)$, where $\phi_0 = \mathbf{h}(\mathbf{d}(\mathbf{x}_0))$.

The first Newton step can be seen as projecting $\Delta\phi_0 = \phi_0 - \phi_*$ onto the row vectors of matrix $\mathbf{R}_0 = -2\mathbf{H}^{-1}\mathbf{J}_\mathbf{h}^\top$. In the rest of the paper, we will refer to $\mathbf{R}_0$ as a *descent direction*. The computation of this descent direction requires the function $\mathbf{h}$ to be twice differentiable or expensive numerical approximations for the Jacobian and Hessian. In our supervised setting, we will directly estimate $\mathbf{R}_0$ from training data by learning a linear regression between $\Delta\mathbf{x}_* = \mathbf{x}_* - \mathbf{x}_0$ and $\Delta\phi_0$. Therefore, our method is not limited to functions that are twice differentiable. However, note that during testing (*i.e.*, inference) ϕ_* is unknown but fixed during the optimization process. To use the descent direction during testing, we **will not use the information of ϕ_* for training**. Instead, we rewrite Eq. 5 as a generic linear combination of feature vector ϕ_0 plus a bias term $\mathbf{b}_0$ that can be learned during training,

$$\Delta\mathbf{x}_1 = \mathbf{R}_0\phi_0 + \mathbf{b}_0. \tag{6}$$

Using training examples, our SDM will learn $\mathbf{R}_0, \mathbf{b}_0$ used in the first step of optimization procedure. In the next section, we will provide details of the learning method.

It is unlikely that the algorithm can converge in a single update step unless f is quadratic under $\mathbf{x}$. To deal with non-quadratic functions, the SDM will generate a sequence of descent directions. For a particular image, the Newton method generates a sequence of updates along the *image-specific* gradient directions,

$$\mathbf{x}_k = \mathbf{x}_{k-1} - 2\mathbf{H}^{-1}\mathbf{J}_\mathbf{h}^\top(\phi_{k-1} - \phi_*). \tag{7}$$

$\phi_{k-1} = \mathbf{h}(\mathbf{d}(\mathbf{x}_{k-1}))$ is the feature vector extracted at previous landmark locations, $\mathbf{x}_{k-1}$. In contrast, SDM will learn a sequence of *generic* descent directions $\{\mathbf{R}_k\}$ and bias terms $\{\mathbf{b}_k\}$,

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{R}_{k-1}\phi_{k-1} + \mathbf{b}_{k-1}, \tag{8}$$

such that the succession of $\mathbf{x}_k$ converges to $\mathbf{x}_*$ for all images in the training set.

3.2. Learning for SDM

This section illustrates how to learn $\mathbf{R}_k, \mathbf{b}_k$ from training data. Assume that we are given a set of face images $\{\mathbf{d}^i\}$ and their corresponding hand-labeled landmarks $\{\mathbf{x}_*^i\}$. For each image starting from an initial estimate of the landmarks $\mathbf{x}_0^i$, $\mathbf{R}_0$ and $\mathbf{b}_0$ are obtained by minimizing the expected loss between the predicted and the optimal landmark displacement under many possible initializations. We choose the L2-loss for its simplicity and solve for the $\mathbf{R}_0$ and $\mathbf{b}_0$ that minimizes

$$\arg\min_{\mathbf{R}_0,\mathbf{b}_0} \sum_{\mathbf{d}^i} \int p(\mathbf{x}_0^i)\|\Delta\mathbf{x}^i - \mathbf{R}_0\phi_0^i - \mathbf{b}_0\|^2 d\mathbf{x}_0^i, \tag{9}$$

where $\Delta\mathbf{x}^i = \mathbf{x}_*^i - \mathbf{x}_0^i$ and $\phi_0^i = \mathbf{h}(\mathbf{d}^i(\mathbf{x}_0^i))$. We assume that $\mathbf{x}_0^i$ is sampled from a Normal distribution whose parameters capture the variance of a face detector. We approximate the integration with Monte Carlo sampling, and instead minimize

$$\arg\min_{\mathbf{R}_0,\mathbf{b}_0} \sum_{\mathbf{d}^i} \sum_{\mathbf{x}_0^i} \|\Delta\mathbf{x}_*^i - \mathbf{R}_0\phi_0^i - \mathbf{b}_0\|^2. \tag{10}$$

Minimizing Eq. 10 is the well-known linear least squares problem, which can be solved in closed-form.

The subsequent $\mathbf{R}_k, \mathbf{b}_k$ can be learned as follows. At each step, a new dataset $\{\Delta\mathbf{x}_*^i, \phi_k^i\}$ can be created by recursively applying the update rule in Eq. 8 with previously learned $\mathbf{R}_{k-1}, \mathbf{b}_{k-1}$. More explicitly, after $\mathbf{R}_{k-1}, \mathbf{b}_{k-1}$ is learned, we update the current landmarks estimate $\mathbf{x}_k^i$ using Eq. 8. We generate a new set of training data by computing the new optimal parameter update $\Delta\mathbf{x}_*^{ki} = \mathbf{x}_* - \mathbf{x}_k^i$ and the new feature vector, $\phi_k^i = \mathbf{h}(\mathbf{d}^i(\mathbf{x}_k^i))$. $\mathbf{R}_k$ and $\mathbf{b}_k$ can be learned from a new linear regressor in the new training set by minimizing

$$\arg\min_{\mathbf{R}_k,\mathbf{b}_k} \sum_{\mathbf{d}^i} \sum_{\mathbf{x}_k^i} \|\Delta\mathbf{x}_*^{ki} - \mathbf{R}_k\phi_k^i - \mathbf{b}_k\|^2. \tag{11}$$

The error monotonically decreases as a function of the number of regressors added. In all our experiments, the algorithm converged in 4 or 5 steps.

3.3. Comparison with existing approaches

A major difference between SDM and discriminative method to fit AAMs [11], is that [11] only uses one step regression, which as shown in our experiments leads to lower performance. Recent work on boosted regression [27, 29, 15, 10] learns a set of weak regressors to model the relation between ϕ and $\Delta\mathbf{x}$. SDM is developed to solve a general NLS problems while boosted regression is a greedy method to approximate the function mapping from ϕ to $\Delta\mathbf{x}$. In the original gradient boosting formulation [16], feature vectors are fixed throughout the optimization, while [15, 10] re-sample the features at the updated landmarks for training different weak regressors. Although they have shown improvements using those re-sampled features, feature re-generation in regression is not well understood and invalidates some properties of gradient boosting. In SDM, the linear regressor and feature re-generation come up naturally in our derivation from Newton method. Eq. 7 illustrates that a Newton update can be expressed as a linear combination of the feature differences between the one extracted at current landmark locations and the template. In previous work, it was unclear what the alignment error function is for discriminative methods. This work proposes Eq. 3, which is the error function minimized by discriminative methods, and connect it with PAMs.

Function	Training Set		Test Set
$h(x)$	$\mathbf{y}$	$\mathbf{x} = h^{-1}(\mathbf{y})$	$\mathbf{y}^*$
$\sin(x)$	[-1:0.2:1]	$\arcsin(\mathbf{y})$	[-1:0.05:1]
x^3	[-27:3:27]	$\mathbf{y}^{\frac{1}{3}}$	[-27:0.5:27]
$\text{erf}(x)$	[-0.99:0.11:0.99]	$\text{erf}^{-1}(\mathbf{y})$	[-0.99:0.03:0.99]
e^x	[1:3:28]	$\log(\mathbf{y})$	[1:0.5:28]

Table 1: Experimental setup for the SDM on analytic functions. $\text{erf}(x)$ is the error function, $\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$.

4. Experiments

This section reports experimental results on both synthetic and real data. The first experiment compares the SDM with the Newton method in four analytic functions. In the second experiment, we tested the performance of the SDM in the problem of facial feature detection in two standard databases. Finally, in the third experiment we illustrate how the method can be applied to facial feature tracking.

4.1. SDM on analytic scalar functions

This experiment compares the performance in speed and accuracy of the SDM against the Newton's method on four analytic functions. The NLS problem that we optimize is:

$$\min_x f(x) = (h(x) - y^*)^2,$$

where $h(x)$ is a scalar function (see Table 1) and y^* is a given constant. Observe that the 1^{st} and 2^{nd} derivatives of those functions can be derived analytically. Assume that we have a fixed initialization $x_0 = c$ and we are given a set of training data $\mathbf{x} = \{x_i\}_{i=1}^n$ and $\mathbf{y} = \{h(x_i)\}_{i=1}^n$. Unlike the SDM for face alignment, in this case no bias term is learned since y^* is known at testing time. We trained the SDM as explained in Sec. 3.2.

The training and testing setup for each function are shown in Table 1 in Matlab notation. We have chosen only invertible functions. Otherwise, for a given y^* multiple solutions may be obtained. In the training data, the output variables $\mathbf{y}$ are sampled uniformly in a local region of $h(x)$, and their corresponding inputs $\mathbf{x}$ are computed by evaluating $\mathbf{y}$ at the inverse function of $h(x)$. The test data $\mathbf{y}^*$ is generated at a finer resolution than in training.

To measure the accuracy of both methods, we computed the normalized least square residuals $\frac{\|\mathbf{x}_k - \mathbf{x}^*\|}{\|\mathbf{x}^*\|}$ at the first 10 steps. Fig. 3 shows the convergence comparison between SDM and Newton method. Surprisingly, SDM converges with the same number of iteration as Newton method but each iteration is faster. Moreover, SDM is more robust against bad initializations and ill-conditions ($f'' < 0$). For example, when $h(x) = x^3$ the Newton method starts from a saddle point and stays there in the following iterations (observe that in the Fig. 3 the Newton method stays at 1). In

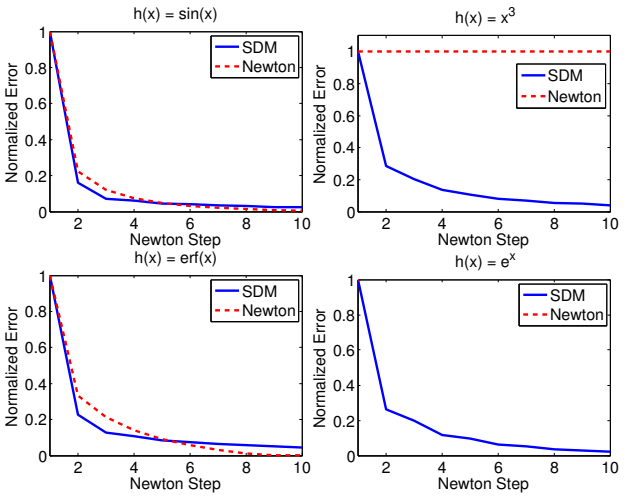


Figure 3: Normalized error versus iterations on four analytic (see Table 1) functions using the Newton method and SDM.

the case of $h(x) = e^x$, the Newton method diverges because it is ill-conditioned. Not surprisingly, when the Newton method converges it provides more accurate estimation than SDM, because SDM uses a generic descent direction. If f is quadratic (*e.g.*, h is linear function of x), SDM will converge in one iteration, because the average gradient evaluated at different locations will be the same for linear functions. This coincides with a well-known fact that Newton method converges in one iteration for quadratic functions.

4.2. Facial feature detection

This section reports experiments on facial feature detection in two "face in the wild" datasets, and compares SDM with state-of-the-art methods. The two face databases are the LFPW dataset[1] [4] and the LFW-A&C dataset [26].

The experimental setup is as follows. First the face is detected using the OpenCV face detector [7]. The evaluation is performed on the images in which a face can be detected. The face detection rates are 96.7% on LFPW and 98.7% on LFW-A&C, respectively. The initial shape estimate is given by centering the mean face at the normalized square. The translational and scaling differences between the initial and true landmark locations are also computed, and their means and variances are used for generating Monte Carlo samples in Eq. 9. We generated 10 perturbed samples for each training image. SIFT descriptors are computed on 32×32 local patches. To reduce the dimensionality of the data, we performed PCA preserving 98% of the energy on the image features.

LFPW dataset contains images downloaded from the web that exhibit large variations in pose, illumination, and facial expression. Unfortunately, only image URLs are given and some are no longer valid. We downloaded 884

[1] http://www.kbvt.com/LFPW/

of the 1132 training images and 245 of the 300 test images. We follow the evaluation metric used in [4], where the error is measured as the average Euclidean distance between the 29 labeled and predicted landmarks. Such error is then normalized by the inter-ocular distance.

We compared our approach with two recently proposed methods [4, 10]. Fig. 4 shows the Cumulative Error Distribution (CED) curves of SDM, Belhumeur *et al.* [4], and our method trained with only one linear regression. Note that SDM is different from the AAM trained in a discriminative manner with linear regression [11], because we do not learn any shape or appearance model (it is non-parametric). Note that such curves are computed from 17 of the 29 points defined in [13], following the convention used in [4]. Clearly, SDM outperforms [4] and linear regression. It is also important to notice that a completely fair comparison is not possible since [4] was trained and tested with more images that were no longer available. However, the average is on favor of our method. The recently proposed method in [10] is based on boosted regression with pose-indexed features. To our knowledge this paper reported the state-of-the-art results on LFPW dataset. In [10], no CED curve is given and they reported a mean error ($\times 10^{-2}$) of 3.43. SDM shows comparable performance with a average of 3.47.

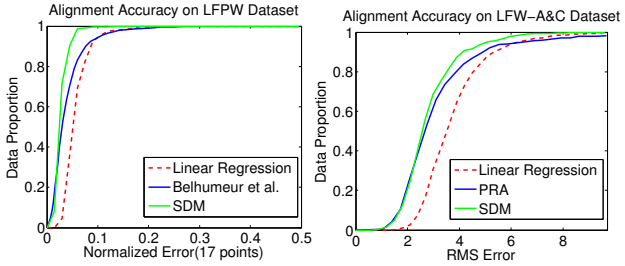
The first two rows of Fig. 6 show our results on the faces with large variations in poses and illumination as well as the ones that are partially occluded. The last row displays the **worst** 10 results measured by the normalized mean error. Most errors are caused by gradient feature's incapability to distinguish between similar facial parts and occluding objects (*e.g.*, glasses frame and eye brows).

LFW-A&C is a subset of LFW dataset[2], consisting of 1116 images of people whose names begin with an 'A' or 'C'. Each image is annotated with the same 66 landmarks shown in Fig. 2. We compared our method with the Principle Regression Analysis (PRA) method [26] that proposes a sample-specific prior to constraint the regression output. This method maintains the state-of-the-art results on this dataset. Following [26], those whose name started with 'A' were used for training giving us a total of 604 images. The remaining images were used for testing. Root mean squared error (RMSE) is used to measure the alignment accuracy. Each image has a fixed size of 250×250 and the error is not normalized. PRA reported a median alignment error of 2.8 on test set while ours averages 2.7. The comparison of CED curves can be found in Fig. 4b and our method outperforms PRA and Linear Regression. Qualitative results from SDM on the more challenging samples are plotted in Fig. 7.

4.3. Facial feature tracking

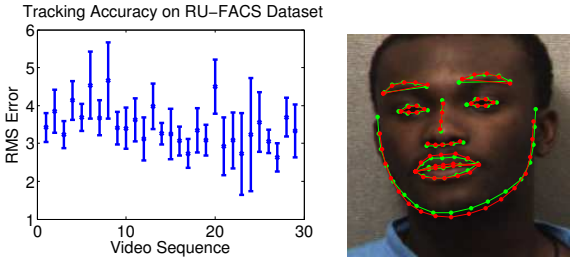
This section tested the use of SDM for facial feature tracking. The main idea is to use SDM for detection in each frame but initializing the frame with the landmark estimate of the previous frame.

[2]http://vis-www.cs.umass.edu/lfw/



(a) (b)

Figure 4: CED curves from LFPW and LFW-A&C datasets.



(a) (b)

Figure 5: a) Average RMS errors and standard deviations on 29 video sequences in RU-FACS dataset. b) RMS error between the SDM detection (green) and ground truth (red) is 5.03.

We trained our model with 66 landmarks on MPIE [17] and LFW-A&C datasets. The standard deviations of the scaling and translational perturbation were set to 0.05 and 10, respectively. It indicates that in two consecutive frames the probability of a tracked face shifting more than 20 pixels or scaling more than 10% is less than 5%. We evaluated SDM's tracking performance on two datasets, RU-FACS [3] and Youtube Celebrities [20].

RU-FACS dataset consists of 29 sequences of different subjects recorded in a constrained environment. Each sequence has an average of 6300 frames. The dataset is labeled with the same 66 landmarks of our trained model except the 17 jaw points that are defined slightly different (See Fig. 5b). We use the remaining 49 landmarks for evaluation. The ground truth is given by a person-specific AAMs [23]. For each of the 29 sequences the average RMS error and standard deviation are plotted in Fig. 5. To make sense of the numerical results, in the same figure we also show one tracking result overlayed with ground truth and in this example it gives us a RMS error of 5.03. We cannot observe obvious differences between the two labelings. Also, the person-specific AAM gives unreliable results when the subject's face is partially occluded while SDM still provides a robust estimation (See Fig. 8). In the 170,787 frames of the RU-FACAS videos, the SDM tracker **never** lost track even in cases of partial occlusion.

Youtube Celebrities is a public "in the wild" dataset[3] that is composed of videos of celebrities during an interview or on a TV show. It contains 1910 sequences of 47 subjects but most of them are less than 3 seconds. It was released as a dataset for face tracking and recognition so no labeled facial landmarks are given. See Fig. 9 for example tracking results from this dataset and tracked video sequences can be found below[4]. From the videos, we can observe that SDM can reliably track facial landmarks with large pose ($\pm 45^\circ$ yaw, $\pm 90^\circ$ roll and, $\pm 30^\circ$ pitch), occlusion and illumination changes. All results are generated without re-initialization. The algorithm is implemented in Matlab/C and tested on an Intel i5-2400 CPU at over 30fps.

5. Conclusions

This paper presents SDM, a method for solving NLS problems. SDM learns in a supervised manner generic descent directions, and is able to overcome many drawbacks of second order optimization schemes, such as non-differentiability and expensive computation of the Jacobians and Hessians. Moreover, it is extremely fast and accurate. We have illustrated the benefits of our approach in the minimization of analytic functions, and in the problem of facial feature detection and tracking. We have shown how SDM outperforms state-of-the-art approaches in facial feature detection and tracking in challenging databases.

Beyond the SDM, an important contribution of this work in the context of algorithms for image alignment is to propose the error function of Eq. 3. Existing discriminative methods for facial alignment pose the problem as a regression one, but lack a well-defined alignment error function. Eq. 3 allows to establish a direct connection with existing PAMs for face alignment, and apply existing algorithms for minimizing it such as Gauss-Newton (or the supervised version proposed in this paper).

In future work, we plan to apply the SDM to other NLS in computer vision such as camera calibration and structure from motion. Moreover, we plan to have a deeper analysis of the theoretical convergence properties of SDM.

Acknowledgements: We would like to thanks R. Cervera and X. Boix for the implementation of the linear and kernel regression method in the fall of 2008. Thanks to J. Saragih for providing the labeled data in experiment 4.2. This work is partially supported by the National Science Foundation (NSF) under the grant RI-1116583 and CPS-0931999. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

[3] http://seqam.rutgers.edu/site/media/data_files/ytcelebrity.tar

[4] http://www.youtube.com/user/xiong828/videos

References

[1] K. T. Abou-Moustafa, F. De la Torre, and F. P. Ferrie. Pareto discriminant analysis. In *CVPR*, 2010. 1

[2] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *IJCV*, 56(3):221 – 255, March 2004. 2

[3] M. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, and J. Movellan. Automatic recognition of facial actions in spontaneous expressions. *Journal of Multimedia*, 1(6):22–35, 2006. 6

[4] P. N. Belhumeur, D. W. Jacobs, D. J. Kriegman, and N. Kumar. Localizing parts of faces using a consensus of exemplars. In *CVPR*, 2011. 2, 3, 5, 6

[5] M. J. Black and A. D. Jepson. Eigentracking: Robust matching and tracking of objects using view-based representation. *IJCV*, 26(1):63–84, 1998. 1, 2

[6] V. Blanz and T. Vetter. A morphable model for the synthesis of 3D faces. In *SIGGRAPH*, 1999. 2

[7] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 5

[8] A. Buchanan and A. W. Fitzgibbon. Damped newton algorithms for matrix factorization with missing data. In *CVPR*, 2005. 1

[9] R. H. Byrd, P. Lu, and J. Nocedal. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific and Statistical Computing*, 16(5):1190–1208, 1995. 2

[10] X. Cao, Y. Wei, F. Wen, and J. Sun. Face alignment by explicit shape regression. In *CVPR*, 2012. 4, 6

[11] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *TPAMI*, 23(6):681–685, 2001. 2, 4, 6

[12] T. F. Cootes, M. C. Ionita, C. Lindner, and P. Sauer. Robust and accurate shape model fitting using random forest regression voting. In *ECCV*, 2012. 3

[13] D. Cristinacce and T. Cootes. Automatic feature localisation with constrained local models. *Journal of Pattern Recognition*, 41(10):3054–3067, 2008. 3, 6

[14] F. De la Torre and M. H. Nguyen. Parameterized kernel principal component analysis: Theory and applications to supervised and unsupervised image alignment. In *CVPR*, 2008. 1, 2

[15] P. Dollár, P. Welinder, and P. Perona. Cascaded pose regression. In *CVPR*, 2010. 2, 4

[16] J. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232, 2001. 2, 4

[17] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. In *AFGR*, 2007. 6

[18] Y. Huang, Q. Liu, and D. N. Metaxas. A component-based framework for generalized face alignment. *IEEE Transactions on Systems, Man, and Cybernetics*, 41(1):287–298, 2011. 3

[19] M. J. Jones and T. Poggio. Multidimensional morphable models. In *ICCV*, 1998. 2

[20] M. Kim, S. Kumar, V. Pavlovic, and H. Rowley. Face tracking and recognition with visual constraints in real-world videos. In *CVPR*, 2008. 2, 6

[21] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 2

[22] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of Imaging Understanding Workshop*, 1981. 1, 2

[23] I. Matthews and S. Baker. Active appearance models revisited. *IJCV*, 60(2):135–164, 2004. 6

[24] S. Rivera and A. M. Martinez. Learning deformable shape manifolds. *Pattern Recognition*, 45(4):1792–1801, 2012. 3

[25] E. Sanchez, F. De la Torre, and D. Gonzalez. Continuous regression for non-rigid image alignment. In *ECCV*, 2012. 3

[26] J. Saragih. Principal regression analysis. In *CVPR*, 2011. 2, 3, 5, 6

[27] J. Saragih and R. Goecke. A nonlinear discriminative approach to AAM fitting. In *ICCV*, 2007. 2, 4

[28] J. Saragih, S. Lucey, and J. Cohn. Face alignment through subspace constrained mean-shifts. In *ICCV*, 2009. 3

[29] P. Tresadern, P. Sauer, and T. F. Cootes. Additive update predictors in active appearance models. In *BMVC*, 2010. 2, 4

[30] G. Tzimiropoulos, S. Zafeiriou, and M. Pantic. Robust and efficient parametric face alignment. In *ICCV*, 2011. 2

[31] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *CVPR*, 2012. 3

[32] K. Zimmermann, J. Matas, and T. Svoboda. Tracking by an optimal sequence of linear predictors. *TPAMI*, 31(4):677–692, 2009. 3

Figure 6: Example results from our method on LFPW dataset. The first two rows show faces with strong changes in pose and illumination, and faces partially occluded. The last row shows the 10 **worst** images measured by normalized mean error.

Figure 7: Example results on LFW-A&C dataset.

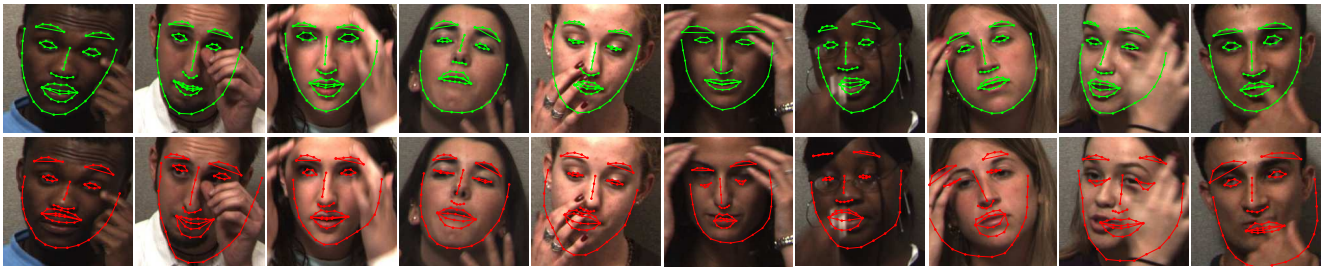


Figure 8: Comparison between the tracking results from SDM (top row) and person-specific tracker (bottom row).

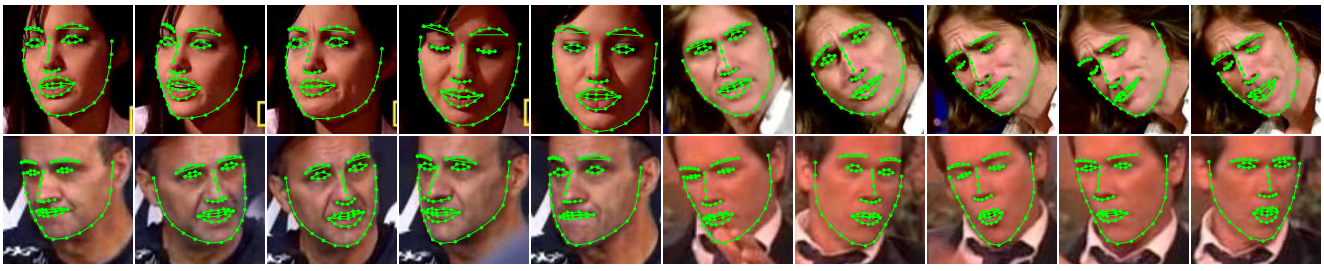


Figure 9: Example results on the Youtube Celebrity dataset.