CSci 6974 and ECSE 6966 Math. Tech. for Vision, Graphics and Robotics
Lectures 23 and 25, April 24 and May 1, 2006
Non-Linear Estimation

Today's Lecture

- We will finish up our discussion of robust estimation and then begin the material on non-linear estimation.
- We will complete the discussion on non-linear estimation on Monday, May 1.
- As a result, these notes are more abbreviated than I initially intended. They focus on gradient descent, Newton's method and Levenberg-Marquardt.
- There will be no homework to be turned in based on these notes, but there are a couple of important practice problems.

Overview

- Motivation and notation
- Problems in 1-d: steepest descent and Newton's method
- Steepest descent in higher dimensions
- Newton's method in higher dimensions
- Levenberg-Marquardt for non-linear least-squares problems
- Application: robust estimation of line/plane parameters
- Application: estimation of plane homographies
- Minimization in 1-d without derivatives
- A final comment on matrix inversion

Motivation and Notation

- Given are a matrix $\mathbf{X}$ of data values and an objective function

$$f(\mathbf{X}; \mathbf{a}) \tag{1}$$

  where $\mathbf{a}$ is the set of parameters to be estimated.

- When f is a quadratic function, which means ∇f (with respect to the parameters) is linear, minimizing this is straight-forward. We've seen this problem several times during the semester.
- For the current set of lecture notes we are interested in the case that f is non-quadratic, which means that ∇f is non-linear.
- Two examples:
  - Each row of $\mathbf{X}$ contains the vector $\mathbf{x}_i^\top, y_i$. The parameter vector is $\mathbf{a}, \sigma$ — it includes the set of line/plane parameters plus σ. The goal is to minimize

$$f(\mathbf{X}; \mathbf{a}, \sigma) = \left\{ \sum_i \rho((y_i - \mathbf{x}_i^\top \mathbf{a})/\sigma) \right\} + N \log \sigma. \tag{2}$$

    This is the robust estimation problem we considered before, but now we have added σ — the robust scale value.
  - $\mathbf{X}$ contains the set of corresponding image points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. The parameter vector is $\mathbf{a} = \mathbf{h}$, a column vector formed from the parameters of plane projective transformation matrix $\mathbf{H}$. The goal is to minimize

$$f(\mathbf{X}; \mathbf{h}) = \sum_i d(\mathbf{x}'_i, \mathbf{H}\mathbf{x}_i)^2 = \sum_i \left(u'_i - \frac{\mathbf{x}_i^\top \mathbf{h}_1}{\mathbf{x}_i^\top \mathbf{h}_3}\right)^2 + \left(v'_i - \frac{\mathbf{x}_i^\top \mathbf{h}_2}{\mathbf{x}_i^\top \mathbf{h}_3}\right)^2 + \tag{3}$$

    In other words, this is the minimization of the geometric error in a single image.
- The solution techniques used here are all iterative. The iterations of the parameter vector estimates will be denoted $\mathbf{a}^t$. We reserve subscripting, such as in a_k, a_l, to denote components of vectors or matrices.

These notes are based more on the *Numerical Recipes in ...* than on Hartley and Zisserman.

Getting Started: One Dimension

1. Steepest descent corresponds to taking the derivative of the function, f, and stepping along the direction of the negative derivative.
2. Newton's method makes a second-order (parabolic) approximation to f, and then finds the minimum of this parabola.
3. There are strengths and weaknesses to each method. What we will be after in the Levenberg-Marquardt technique is a unification of the methods that preserves the strengths of each.

Approximating the Function in Multiple Dimensions

We take a Taylor expansion:

$$f(\mathbf{X};\mathbf{a}+\Delta\mathbf{a}) \approx f(\mathbf{X};\mathbf{a}) + \nabla f(\mathbf{X};\mathbf{a})\Delta\mathbf{a} + \frac{1}{2}\Delta\mathbf{a}^\top \mathbf{D}(\mathbf{X};\mathbf{a})\Delta\mathbf{a} \tag{4}$$

where $\mathbf{D}$ is the Hessian. (I've used $\mathbf{D}$ instead of $\mathbf{H}$ to avoid confusion with the projective transformation matrix $\mathbf{H}$.)

Steepest Descent

- Starting from an initial estimate, $\mathbf{a}^0$, each successive estimate is computed as
$$\mathbf{a}^{t+1} = \mathbf{a}^t - c^t \cdot \nabla f(\mathbf{X};\mathbf{a}^t)^\top \tag{5}$$
where the c^t is a small constant.
- Determining c^t is problematic, and c^t must often be adjusted empirically.
- Iterations stop when $\nabla f(\mathbf{X};\mathbf{a}^t)$ is sufficiently small.

Newton's Method

- Taking the derivative of the approximation (4), setting the result equal to $\mathbf{0}$, and solving yields the iterative equation:
$$\mathbf{a}^{t+1} = \mathbf{a}^t - \mathbf{D}(\mathbf{X};\mathbf{a}^t)^{-1}\nabla f(\mathbf{X};\mathbf{a}^t)^\top \tag{6}$$
- This fails if the resulting $\mathbf{a}^{t+1}$ does not reduce the value of the objective function.

Strengths and Weaknesses → Levenberg-Marquardt

- Steepest descent is slow to converge and it is hard to determine the step size. But, no computation of the Hessian is needed.
- Newton's method has faster (quadratic) convergence, but requires computation of the Hessian and doesn't handle poorly-behaved functions.
- Levenberg-Marquardt is a hybrid of these for nonlinear least-squares problems with an automatically-adjusted parameter that switches back-and-forth between them.
- We will assume the objective function is close to the form
$$f(\mathbf{X};\mathbf{a}) = \sum_{\mathbf{x}_i \in \mathbf{X}} \left[e_i(\mathbf{x}_i;\mathbf{a})\right]^2. \tag{7}$$
where e_i is some form of error distance.

A Simplified Hessian for Levenberg-Marquardt

- Each entry in the Hessian is of the form

$$\frac{\partial^2 f}{\partial a_k \partial a_l} = \sum_{\mathbf{x}_i} 2\left[\frac{\partial e_i}{\partial a_k}\frac{\partial e_i}{\partial a_l} + e_i \frac{\partial^2 e_i}{\partial a_k \partial a_l}\right] \tag{8}$$

  where I've shortened $e_i(\mathbf{x}_i; \mathbf{a})$ to just e_i for notational convenience. Observe that it is a combination of first-order and second-order derivative terms.

- Most L-M implementations drop the second-order derivative terms from the Hessian. There are several reasons for this:
  - Near the solution, where the Hessian really matters, e_i should be small.
  - The e_i should be small random errors and therefore cancel each other out, especially near the minimum.
  - The second partials are relatively unstable far from the solution.
  - Dropping the second partials saves dramatically on the pain of computation.

- As a result, we will compute the k, l term of the Hessian as

$$D_{k,l} = \sum_{\mathbf{x}_i} 2\frac{\partial e_i}{\partial a_k}\frac{\partial e_i}{\partial a_l} \tag{9}$$

- Finally, if you have an implementation of the function, but don't have (or don't want to calculate) an implementation of the derivative, numerical differentiation can be used.

Levenberg-Marquardt (L-M)

We'll give mathematical details first:

- Rewrite the Hessian slightly, but only for the diagonal terms:

$$D_{k,k} = (1+\lambda) \sum_{\mathbf{x}_i \in \mathbf{X}} 2\left(\frac{\partial e_i}{\partial a_k}\right)^2 \tag{10}$$

  Here, λ is an adaptive parameter.

- The L-M algorithm to find a local minimum of f, starting from an initial parameter estimate $\mathbf{a}^0$, is as follows:
  1. Initialize $\lambda = 0.001$ or some other suitably small parameter.
  2. $t = 1$

3. Repeat the following steps until convergence:
   (a) Compute ∇f and $\mathbf{D}$ using the current value of λ.
   (b) Assign $\Delta\mathbf{a} = -\mathbf{D}^{-1}\nabla f^{\top}$.
   (c) If $f(\mathbf{X}; \mathbf{a}^{t-1} + \Delta\mathbf{a}) < f(\mathbf{X}; \mathbf{a}^{t-1})$ then assign
      - $\mathbf{a}^t = \mathbf{a}^{t-1} + \Delta\mathbf{a}$,
      - $\lambda = \lambda/10$.
   (d) else
      - $\mathbf{a}^t = \mathbf{a}^{t-1}$
      - $\lambda = \lambda \cdot 10$.
   (e) $t = t + 1$

- Convergence is obtained when the objective function changes by a negligible amount (one or several times) for updated values of $\mathbf{a}^t$.

Levenberg-Marquardt: Justification

- Adds a measure of dimensionality to the steepest descent technique.
- Acts as a trade-off between steepest descent and Newton's method through the λ parameter.
- The λ parameter may be viewed as conditioning the Hessian matrix.

Our Two Applications

- Robust estimation of line/plane parameters. The L-M implementation still requires a good initialization.
- Plane homographies. We will look at one of the simpler objective functions. See Hartley & Zisserman for harder ones.

Towards Additional Techniques: Minimization Along 1-D

Subgoal: given a direction $\boldsymbol{\theta}$ in the space of $\mathbf{a}$ and given the current estimate, $vecta^t$, find the minimum along the direction $\boldsymbol{\theta}$. In other words, find r such that

$$f(\mathbf{X}; \mathbf{a}^t + r\boldsymbol{\theta}) \tag{11}$$

is minimized.

- If we can compute gradients, we can convert the gradient to a directional derivative and steepest descent can be used.
- If we can compute gradients and the Hessian, Newton's method can be used.

- If we don't want to or can't compute derivatives then we can use a "bracketing" Golden-Section search technique similar in spirit to a binary or bisection search. This technique involves three locations along $\boldsymbol{\theta}$: q, r, s such that $q < r < s$ and $f(\mathbf{X}; \mathbf{a}^t + q\boldsymbol{\theta}) > f(\mathbf{X}; \mathbf{a}^t + r\boldsymbol{\theta})$, $f(\mathbf{X}; \mathbf{a}^t + r\boldsymbol{\theta}) < f(\mathbf{X}; \mathbf{a}^t + s\boldsymbol{\theta})$. The search sections the interval $[q \dots s]$ until $s - q$ is sufficiently small.

- These one-dimensional minimization methods are often used in combination with other methods for multidimensional problems. For example, you might

  1. Compute (perhaps numerically) the gradient direction at the initial estimate,
  2. Minimize along this direction using one of the above techniques, then
  3. Choose a new direction, complementary to the original direction, and repeat.

Final Comment on Matrix Inversion

The solution to many problems involves inverting a large matrix, solving

$$\mathbf{A}\mathbf{x} = \mathbf{y}. \tag{12}$$

If the dimensions of $\mathbf{A}$ are large, one must be careful and choose the method according to the structure of $\mathbf{A}$:

- If $\mathbf{A}$ is sparse (e.g. banded), then choose QR or LU methods, since SVD makes a dense matrix from a sparse one.

- If $\mathbf{A}$ is dense then the SVD makes sense.

- If the dimensions of $\mathbf{A}$ grow extreme, then numerical inversion techniques must be used.

Practice Problems

1. Develop a Levenberg-Marquardt technique to estimate the parameters of a conic based on minimizing the Sampson error. Start by showing how to generate an initial estimate using a normalization technique together with the algebraic error. Then write-out the error distance. Finally, derive the expressions for the gradient and Hessian.

2. Show how to adapt our description of Newton's method to find the root of an equation in one dimension. Stated another way, given function $f(x)$ and initial value x_0, find a value x' such that $f(x') = 0$.