Lecture 9: Local Optimization

1 The Maximal Likelihood Estimator

In the previous lecture we derived the maximum likelihood estimator for our class of projection problems. If $x_{ij} = (x_{ij}^1, x_{ij}^2)$ is the projection in regular Cartesian coordinates of the 3D-point $\mathbf{X}_j$ in camera P_i then the model parameters that make the measurements x_{ij} most likely is found by minimizing

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|\left(x_{ij}^1 - \frac{P_i^1\mathbf{X}_j}{P_i^3\mathbf{X}_j},\ x_{ij}^2 - \frac{P_i^2\mathbf{X}_j}{P_i^3\mathbf{X}_j}\right)\right\|^2. \tag{1}$$

This problem is however difficult to solve since there is in general no closed form solution. In practice we are limited to locally improving the objective function from some suitable selected starting point. The starting point is typically generated using algebraic solvers like the ones we have presented previously in the course. In this lecture we will present some simple methods for local optimization.

2 Background

We first recall some basic concepts from optimization.

Definition 1. *A point v is called a stationary point of f if*

$$\nabla f(v) = 0. \tag{2}$$

Definition 2. *A point v is called a local minimizer of f if*

$$f(v) \leq f(s) \tag{3}$$

for all s in a neighborhood around v. (More technically, for all s such that $\|v - s\| \leq \delta$ for some $\delta > 0$).

Theorem 1. *Any local minimizer of a differentiable function is also a stationary point.*

Definition 3. *A matrix M is positive definite ($M \succ 0$) if $v^T M v > 0$ for all $v \neq 0$. It is called positive semi definite ($M \succeq 0$) if $v^T M v \geq 0$ for all v.*

Theorem 2. *If v is a stationary point of f ($\nabla f(v) = 0$) and the hessian of f is positive definite at t ($\nabla^2 f(v) \succ 0$) then v is also a local minima of f.*

3 Linear Least Squares

Minimizing (1) is an example of a non-linear least squares problem. In this section we will show how linear least squares problems can be minimized. In in the following sections we will use this approach to tackle the more difficult non-linear versions.

In the linear least squares problem we want to fit a set of linear functions $A_i v$ to a set of measurements b_i, $i = 1, .., n$. Here A_i^T is a vector of the same size as v representing a linear function of v. The least squares problem is now

$$\min_v \sum_{i=1}^{n} \|A_i v - b_i\|^2. \tag{4}$$

If we concatenate all the A_i into one matrix A and all the b_i into one vector b we can write the problem in matrix form as

$$\min_v \|Av - b\|^2. \tag{5}$$

Expanding the norm gives

$$\|Av - b\|^2 = (Av - b)^T(Av - b) = \underbrace{v^T A^T Av - 2b^T Av + b^T b}_{=f(v)}. \tag{6}$$

To find the minimum of this function we compute its stationary points.

$$\nabla f(v) = 0 \Leftrightarrow 2A^T Av - 2A^T b = 0 \Leftrightarrow A^T Av = A^T b. \tag{7}$$

These equations are called the normal equations, and assuming that $A^T A$ is invertible they give the stationary point

$$v = (A^T A)^{-1} A^T b. \tag{8}$$

To see that this is a local minimizer we look at the Hessian of f

$$\nabla^2 f(v) = A^T A. \tag{9}$$

This matrix is positive semi definite since

$$v^T A^T Av = \|Av\|^2 \geq 0. \tag{10}$$

Furthermore, it has to be positive definite since if there is $v \neq 0$ such that $\|Av\| = 0$ then, by multiplying with A^T, we see that $A^T Av = 0$ which would imply that $A^T A$ is not invertible (which we have assumed above). Therefore, (8) is a local minimum.

It is in fact also a global minimum which can be seen by changing coordinates. Let us call the stationary point s and let $v = \delta + s$. We then get

$$\|A(\delta + s) + b\|^2 = (\delta + s)^T A^T A(\delta + s) - 2b^T A(\delta + s) + b^T b. \tag{11}$$

Since s is the stationary point it fulfills (7) and therefore the right hand side can be re-written as

$$(\delta + s)^T A^T A(\delta + s) - 2s^T A^T A(\delta + s) + b^T b = \underbrace{\delta^T A^T A\delta}_{\text{depends on } \delta} - \underbrace{s^T A^T As + b^T b}_{\text{const.}}. \tag{12}$$

The result of this simplification is a term which depends on δ and one that is constant. Since $A^T A$ is positive definite any choice of δ that is not zero will result in the first term being positive. Therefore the optimal choice is $\delta = 0$ and thereby $v = s$.

4 Non-Linear Least Squares

Next we turn to the problem of solving the non-linear least squares formulation. In the general formulation we have a set of residuals $r_i(v)$ which we want to minimize in a least squares sense

$$\min_v \sum_i r_i(v)^2. \tag{13}$$

If we stack all the residuals $r_i(v)$ in a vector $r(v)$ then we can write the problem

$$\min_v \|r(v)\|^2. \tag{14}$$

In this section we will present three simple procedures that from a starting point improves the objective value by locally searching for better values.

4.1 Steepest Descent

The perhaps simplest possible strategy is the **steepest descent** search. Given a starting point v_0 this method finds the direction in which the function decreases most rapidly, and takes a step in this direction. For the function $f(v)$ the directional derivatives at a point v_0 are given by

$$f'_d(v_0) = \nabla f(v_0)^T d. \tag{15}$$

Here d is a direction (vector of unit length). To find the d for which $f'_d(v_0)$ is maximally negative we should select $d = -\nabla f(v_0)/|\nabla f(v_0)|$ since this choice would give

$$f'_d = -\nabla f(v_0)^T \frac{\nabla f(v_0)}{|\nabla f(v_0)|} = -|\nabla f(v_0)|. \tag{16}$$

For each term $r_i(v)^2$ of the function $f(v) = \|r(v)\|^2 = r_1(v)^2 + r_2(v)^2 + ... + r_n(v)^2$ we have

$$\nabla \left(r_i(v)^2\right) = 2r_i(v)\nabla r_i(v), \tag{17}$$

and therefore

$$\nabla f(v) = 2\sum_i r_i(v)\nabla r_i(v). \tag{18}$$

Once the descent direction d has been computed the algorithm picks a new point v_1 by searching along this direction, that is $v_1 = v_0 + \lambda d$, where lambda is selected such that $f(v_1) < f(v_0)$. It is always possible to find such a λ (buy choosing λ small) unless $\nabla f(v_0) = 0$, that is v_0 is already a stationary point.

Then the whole process is repeated for the point v_1 and so on.

4.2 Gauss-Newton

While very cheap to compute the gradient does not contain any information about step-length. Therefore in the steepest descent approach we are forced to select a suitable λ by other means.

An alternative approach is the Gauss-Newton method. In this method we first approximate the residuals $r_i(v)$ with linear functions and then solve the resulting approximation using the approach from Section 3. The Taylor approximation of $r(v)$ at v_0 is

$$r(v) = \begin{pmatrix} r_1(v) \\ r_2(v) \\ \vdots \\ r_n(v) \end{pmatrix} \approx \underbrace{\begin{pmatrix} r_1(v_0) \\ r_2(v_0) \\ \vdots \\ r_n(v_0) \end{pmatrix}}_{r(v_0)} + \underbrace{\begin{pmatrix} \nabla r_1(v_0)^T \\ \nabla r_2(v_0)^T \\ \vdots \\ \nabla r_n(v_0)^T \end{pmatrix}}_{J(v_0)} \underbrace{(v - v_0)}_{d} \tag{19}$$

The matrix $J(v_0)$ is the Jacobian of the vector r. Its rows contain the gradients of each entry r_i in r. The approximating linear least squares problem is therefore

$$\min_d \|r(v_0) + J(v_0)d\|^2, \tag{20}$$

which according to (8) has the solution

$$d = -(J(v_0)^T J(v_0))^{-1} J(v_0)^T r(v_0). \tag{21}$$

The Gauss-Newton update is now $v_1 = v_0 + d$.

Remark 1. Note that with the notation in this section the steepest descent direction can (if we ignore the normalization) be written

$$d = -J(v_0)^T r(v_0). \tag{22}$$

4.3 Levenberg-Marquardt

While convergence of the Gauss-Newton method is very rapid close to the minimum it can be unstable at points not sufficiently close to the minimum. The reason is that the approach may generate large steps even though the approximation is only valid in a local neighborhood around v_0. Therefore we add a penalty for large steps and solve

$$\min_d \|r(v_0) + J(v_0)d\|^2 + \lambda\|d\|^2. \tag{23}$$

To find the minimum of this problem we can again compute the stationary point. Expanding the function gives

$$\|r(v_0) + J(v_0)d\|^2 + \lambda\|d\|^2 = d^T \left(J(v_0)^T J(v_0) + \lambda I\right) d + 2r(v_0)^T J(v_0)d + r(v_0)^T r(v_0). \tag{24}$$

Differentiating with respect to d gives

$$2\left(J(v_0)^T J(v_0) + \lambda I\right) d + 2J(v_0)^T r(v_0) = 0, \tag{25}$$

and therefore

$$d = -\left(J(v_0)^T J(v_0) + \lambda I\right)^{-1} J(v_0)^T r(v_0). \tag{26}$$

This approach is called the Levenberg-Marquardt update and is often much more stable than the original Gauss-Newton formulation. If λ is selected large enough the update $v_1 = v_0 + d$ is guaranteed to give a better objective value. On the other hand for a very large λ (26) is almost the same as (22) (multiplied with $1/\lambda$). A common strategy is to start with a large λ and gradually reducing it to increase convergence speed when approaching the minimum value.

4.4 Bundle Adjustment

We now return to the structure from motion problem. We will consider the calibrated version, that is, we are assuming that the image points x_{ij} have already been normalized and that we are searching for camera matrices of the form $P_i = [R_i \quad t_i]$ and scene points or the form $\mathbf{X}_j = \begin{bmatrix} X_j \\ 1 \end{bmatrix}$. In Computer Vision literature this problem is often called bundle adjustment since the bundle of viewing rays are locally adjusted to reduce the reprojection errors. The objective function (1) can then be written

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \left\| \left(x_{ij}^1 - \frac{R_i^1 X_j + t_i^1}{R_i^3 X_j + t_i^3},\; x_{ij}^2 - \frac{R_i^2 X_j + t_i^2}{R_i^3 X_j + t_i^3} \right) \right\|^2, \tag{27}$$

where R_i^1, R_i^2, R_i^3 are the rows of the rotation R_i and t_i^1, t_i^2, t_i^3 are the entries of the translation t_i. Note that we have assumed in (27) that all points are visible in all cameras. However, this is no restriction. If for example, point 2 is not visible in camera 3 then we simply remove the term $(i, j) = (3, 2)$ from the sum.

To be able to apply the Gauss-Newton method we need to linearize the expressions of the type

$$\underbrace{\frac{R^1 X + t^1}{R^3 X + t^3}}_{=f_1} \text{ and } \underbrace{\frac{R^2 X + t^2}{R^3 X + t^3}}_{=f_2}. \tag{28}$$

(For ease of notation we drop the indexes i, j for now.) There are two difficulties with this that we need to handle: First, the functions are nonlinear since they contain both fractions and quadratic terms. Second, the rotation R depends non-linearely on a set of parameters.

One way to parametrize a rotation is through the exponential map

$$\exp(A) = \sum_{k=0}^{\infty} \frac{1}{k!} A^k = I + A + \frac{1}{2}A^2 + \frac{1}{6}A^3 + ... \tag{29}$$

If R_0 is our current rotation estimate, then any other rotation R can be written as R_0 multiplied with a the exponential map of a skew symmetric matrix

$$R = \exp \begin{pmatrix} 0 & -a_1 & -a_2 \\ a_1 & 0 & -a_3 \\ a_2 & a_3 & 0 \end{pmatrix} R_0. \tag{30}$$

Since we are interested in linearizing (28) we need only consider the first order terms of (29). If we let

$$S_1 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad S_2 = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad \text{and} \quad S_3 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}, \tag{31}$$

then close to R_0 we have

$$R \approx (I + (a_1 S_1 + a_2 S_2 + a_3 S_3)) R_0. \tag{32}$$

This gives us a linear local parametrization of the rotation in terms of the variables a_1, a_2, a_3. To determine the linearization of (28) we need to compute derivatives with respect to all the variables $a_1, a_2, a_3, t^1, t^2, t^3, X_1, X_2$ and X_3 in the current parameter estimates R_0, t_0, X_0. This is straight forward (although tedious) and we only give a few of the derivatives here:

$$\frac{\partial f_1}{\partial a_1} = \frac{S_1^1 R_0 X_0}{R_0^3 X_0 + t_0^3} - \frac{R_0^1 X_0 + t_0^1}{(R_0^3 X_0 + t_0^3)^2} S_1^3 R_0 X_0 \tag{33}$$

$$\frac{\partial f_1}{\partial X_0^1} = \frac{R_0^{11}}{R_0^3 X_0 + t_0^3} - \frac{R_0^1 X_0 + t_0^1}{(R_0^3 X_0 + t_0^3)^2} R_0^{31} \tag{34}$$

$$\frac{\partial f_1}{\partial t^1} = \frac{1}{R_0^3 X_0 + t_0^3} \tag{35}$$

$$\frac{\partial f_1}{\partial t^2} = 0 \tag{36}$$

$$\frac{\partial f_1}{\partial t^3} = -\frac{R_0^1 X_0 + t_0^1}{(R_0^3 X_0 + t_0^3)^2}, \tag{37}$$

where S_1^i is the i'th row of S_1, $R_0^{i,j}$ is element (i, j) in R_0. Using these (and the rest of the) derivatives we can now form the Jacobian $J(v_0)$ in (19). The entries of $r(v_0)$ is obtained by computing the residual values at the current parameter estimate. Note that v_0 contain the parameter values of $a_1, a_2, a_3, t^1, t^2, t^3, X_1, X_2$ and X_3 for all the cameras and all the points of the problem. Thus if there are n cameras, m scene points and all the points are visible in all the cameras then v_0 is of size $(6n + 3m) \times 1$, $r(v_0)$ is $mn \times 1$ and $J(v_0)$ is $(6n + 3m) \times mn$.

Since the reconstruction is only uniquely determined up to an unknown similarity transformation we can reduce the number of variables slightly. For example we can assume that the first camera is $[I \quad 0]$. This fixes six of the seven degrees of freedom of the similarity transformation. To also fix the scale we can for example select the first coordinate of the first point.