Lecture 18: Surrogate Functions and the MM Algorithm

Chapter 8.1–8.2

Charles A. Bouman

Purdue University

Monday, October 5, 2026

Outline

1. **Motivation**: nonquadratic cost, majorization, surrogate
2. **The two conditions**: equal at the current point, above it elsewhere
3. **MM algorithm**: update, inequality chain, monotone decrease
4. **Definition**: the shift $c(x')$ and the formal definition

Reading: Chapter 8.1–8.2, *Foundations of Computational Imaging*.

Today's Big Ideas

- A surrogate touches the cost at the current point, above it elsewhere
- Minimizing the surrogate can only lower the true cost.
- Repeating this lowers the cost each step: the MM algorithm

The nonquadratic MAP cost function

From Chapter 7, the MAP cost function for a linear forward model with a non-Gaussian MRF prior is

$$f(x;y) = \frac{1}{2}||y - Ax||_{\Lambda}^2 + \sum_{\{s,r\}\in\mathcal{P}} b_{s,r}\rho(x_s - x_r) \ .$$

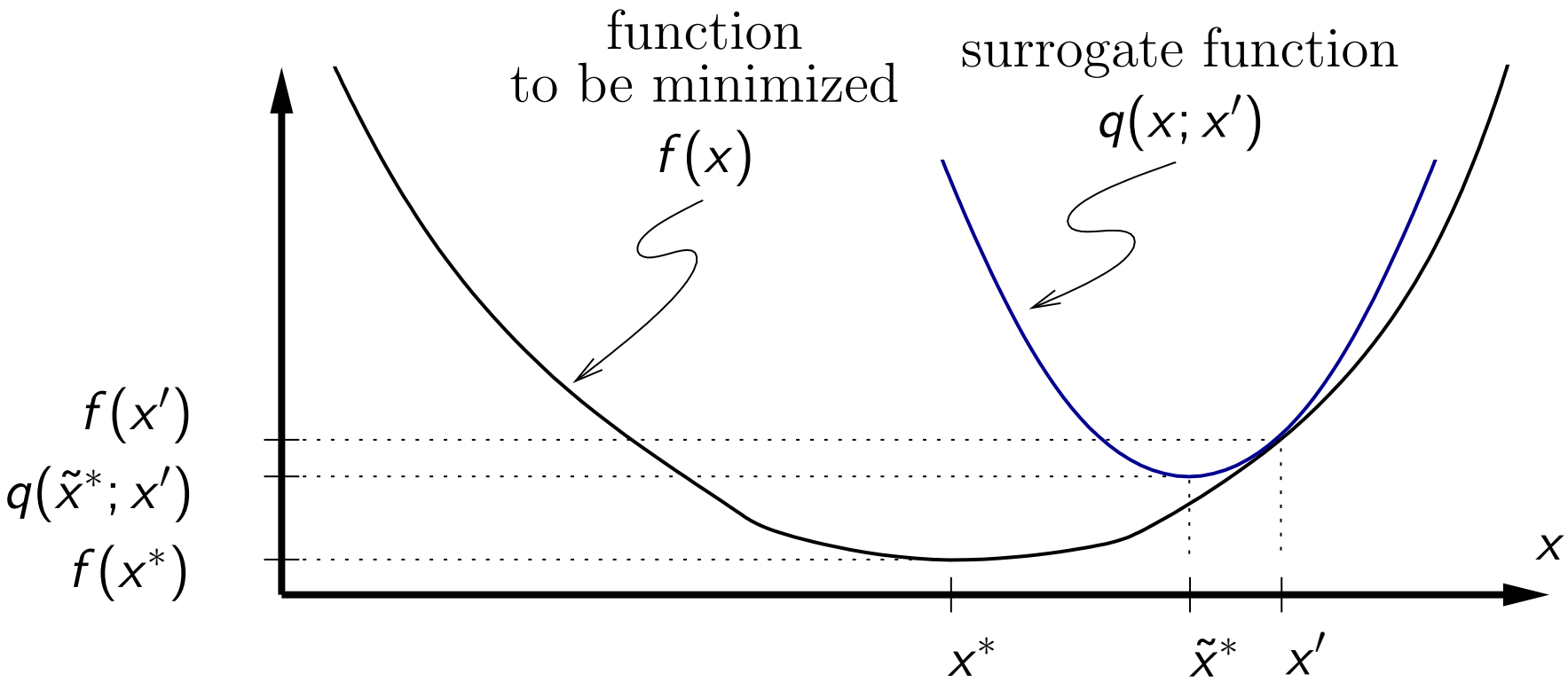
- The **data term** is quadratic, so Chapter 5 methods already handle it
- The **prior term** is not quadratic when ρ is not quadratic.
- Data y is fixed in Chapter 8, so we write the cost as $f(x)$

Majorization and the surrogate function

Majorization techniques convert nonquadratic optimization problems into iterative quadratic optimization.

- The cost $f(x)$ is complicated, so direct minimization is hard
- Instead, minimize a simpler **surrogate function** q
- q fits f near the **point of approximation**, like a Taylor series

The surrogate function, illustrated



Both $f(x)$ and $q(x; x')$ are plotted against x, and the two are equal at the point of approximation $x = x'$. The point x^* is the global minimum of $f(x)$, and $\tilde{x}^*$ is the global minimum of $q(x; x')$.

The two conditions on the surrogate function

We insist that the surrogate function satisfy two conditions:

The two conditions on $q(x; x')$

$$f(x') = q(x'; x') \ , \tag{8.1}$$
$$f(x) \leq q(x; x') \ . \tag{8.2}$$

- Condition (8.1): the two functions are equal at x'
- Condition (8.2): q is an upper bound everywhere else

The majorization minimization algorithm

We cannot easily minimize the complicated cost $f(x)$ directly. But at the current point $x^{(k)}$ we can build the simpler surrogate $q(x; x^{(k)})$ and minimize that instead.

Take its minimizer as the next point, then repeat. This is the **majorization minimization (MM) algorithm**.

The majorization minimization update

$$x^{(k+1)} = \arg\min_{x \in \mathbb{R}^N} \left\{ q(x; x^{(k)}) \right\} \ . \tag{8.3}$$

Why minimizing q must reduce f

Each step can only lower the cost, or leave it the same. This chain of inequalities shows why:

The inequality chain

$$f(x^{(k+1)}) \leq q(x^{(k+1)}; x^{(k)}) \leq q(x^{(k)}; x^{(k)}) = f(x^{(k)}) \ . \tag{8.4}$$

- The **first inequality** is the upper bound (8.2), $f(x) \leq q(x; x')$.
- The **second inequality** is the minimization (8.3).
- The **third equality** is condition (8.1), $f(x') = q(x'; x')$.

Monotone decrease and convergence

- Chain (8.4) holds each step, so $f(x^{(k)})$ is **monotone decreasing**
- If the cost is bounded below, the cost values **converge**

The iterates themselves need not converge

A common misconception is that the points $x^{(k)}$ must also converge. This is usually true in practice, but it is **not guaranteed** by conditions (8.1), (8.2), and (8.3).

Relaxing the conditions: the shift $c(x')$

We can relax the conditions and still keep the desired outcome. Let $c(x')$ be any function of the current state x', and define

$$Q(x; x') = q(x; x') + c(x') \ .$$

Minimizing $Q(x; x')$ gives the same update as (8.3), so Q can be used in place of q.

Given $Q(x; x')$, we recover $q(x; x')$ by shifting:

Recovering q from Q

$$q(x; x') = Q(x; x') - Q(x'; x') + f(x') \ . \tag{8.5}$$

This shift enforces (8.1), that $f(x') = q(x'; x')$. Substituting (8.5) into (8.2) gives the definition of a valid surrogate function.

Definition: surrogate function

Surrogate Function

Let $f(x)$ be a function on $\mathcal{A} \subset \mathbb{R}^N$. Then $Q(x; x')$ is a surrogate function for **minimizing** $f(x)$ if for all $x, x' \in \mathcal{A}$,

$$f(x) \leq Q(x; x') - Q(x'; x') + f(x') .$$

Similarly, $Q(x; x')$ is a surrogate function for **maximizing** $f(x)$ if for all $x, x' \in \mathcal{A}$,

$$f(x) \geq Q(x; x') - Q(x'; x') + f(x') .$$

- We treat minimization; for maximization, negate the function

Pseudocode: majorization minimization algorithm

Majorization Minimization Algorithm:

Initialize x'

For $k = 1$ to K {

$$x' \leftarrow \arg\min_{x \in \mathcal{A}} Q(x; x')$$

}

$x^* = x'$

In each iteration the true function $f(x)$ is approximated by a surrogate function $Q(x; x')$, and the surrogate is then minimized. Each new surrogate is built at the value of x' from the previous iteration.

Summary

- A surrogate equals the cost at the current point, above it elsewhere
- The MM algorithm builds a surrogate, minimizes it, and repeats.
- Each step lowers the cost or leaves it unchanged.
- A shift that depends only on the current point leaves the update unchanged. This gives the formal definition of a surrogate.

Reading: Chapter 8.1–8.2. **Next lecture:** properties of surrogate functions.