Signal Denoising

Consider the task of removing noise from a one-dimensional signal (e.g. a time signal) given by the data set $d = \{d_k\}_{k=1}^n$. A common formulation is to find the related set $u = \{u_k\}_{k=1}^n$ that is the minimizer of the following optimization problem.

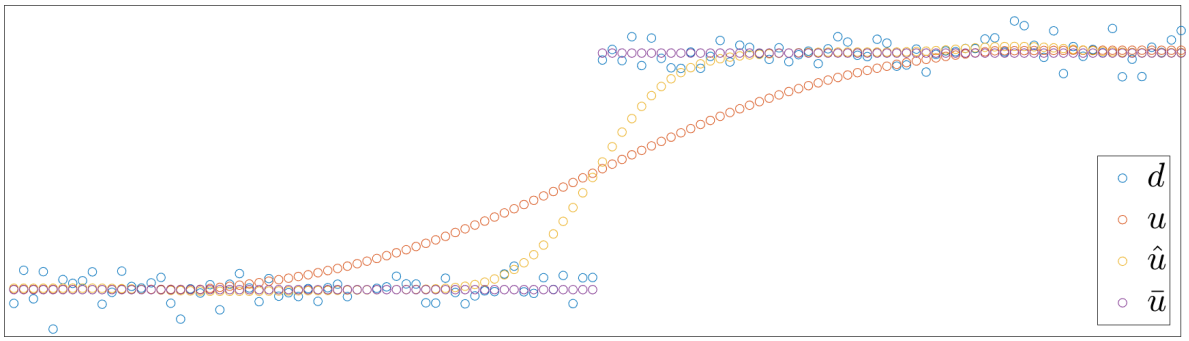
$$\min_u \quad f(u) = \frac{1}{2}\sum_{k=1}^{n}(u_k - d_k)^2 + \frac{\alpha}{2}\sum_{k=1}^{n-1}(u_{k+1} - u_k)^2 .$$

The optimal result $u^* = \{u_k^*\}$ represents a "denoised" signal that is smoother than the data (second term) but remains similar to the data (first term). The degree to which these goals are attained is governed by the choice of parameter α. We have seen that this process will smooth out not only noise but any sharp discontinuities as well. So, if we expect our denoised signal to accurately represent discontinuities, then we will need a different approach.

Consider instead the generalized optimization problem

$$\min_u \quad f(u) = \frac{1}{2}\sum_{k=1}^{n}(u_k - d_k)^2 + \frac{\alpha}{p}\sum_{k=1}^{n-1}|u_{k+1} - u_k|^p , \tag{9}$$

where the second term in the objective allows for adjustable variation regularization through the parameter p. We have seen that for $p = 2$, u^* is biased toward smooth variations and for $p = 1/2$, u^* is biased toward discontinuous variations. For $p = 1$ any monotonic variation is equally penalized. This situation can be seen in the figure below in which each of the data sets u, $\hat{u}$ and $\bar{u}$ have identical $p = 1$ total variation. However, clearly $\bar{u}$ more closely approximates the data and so evaluates to a lower total objective value.



The potential difficulty with optimization problem (9) is that the objective is nonsmooth for $p \leq 1$. Thus, the use of gradient-based methods of solution may behave poorly and are not guaranteed to find a local minimizer. One effective and common approach is to regularize the total variation term using a small parameter β:

$$\min_u \quad f(u) = \frac{1}{2}\sum_{k=1}^{n}(u_k - d_k)^2 + \frac{\alpha}{p}\sum_{k=1}^{n-1}\left(\sqrt{(u_{k+1} - u_k)^2 + \beta^2}\right)^p . \tag{10}$$

For β much less than the typical variation in the data, the second term in (10) closely approximates the variation in u and is a smooth function of u. Finally, we can improve this

method by incorporating optional boundary conditions. That is, suppose we wish enforce $u_0 = L$ and $u_{n+1} = R$. Then we have the objective function (note the limits on the second sum)

$$\min_u \quad f(u) = \frac{1}{2}\sum_{k=1}^{n}(u_k - d_k)^2 + \frac{\alpha}{p}\sum_{k=0}^{n}\left(\sqrt{(u_{k+1}-u_k)^2 + \beta^2}\right)^p \tag{11}$$

Keep in mind that u_0 and u_{n+1} are not decision variables – they are constants supplied by the user. We will use this formulation to remove noise from signal data. The gradient of f can be computed (from Equation 11) using the following observation:

$$\begin{aligned}\frac{\partial f}{\partial u_k} = (u_k - d_k) &+ \alpha\left[(u_{k+1}-u_k)\left(\sqrt{(u_{k+1}-u_k)^2+\beta^2}\right)^{p-2}\right] \\ &- \alpha\left[(u_k - u_{k-1})\left(\sqrt{(u_k-u_{k-1})^2+\beta^2}\right)^{p-2}\right]\end{aligned}$$

Note: The form of the gradient is equivalent to that given in class, but has a simpler representation. You should feel free to use this form for all α, β and p. However, your code will be more efficient if you have a separate case for $p = 2$. And finally, if you do not employ boundary conditions for a particular data set, then the partial derivatives for $k = 1$ and at $k = n$ will be slightly different than shown above.

Complete the following tasks.

1. Create a function that computes the total variation objective value given a set of input data values d, a corresponding set of candidate denoised values u, and parameters α, β, p and optional boundary conditions $u_0 = L$ and $u_{n+1} = R$. Also output an exact gradient (not computed with finite differences).

2. Conjecture methods for choosing appropriate values of α, β and p based on the properties of the data set d.

3. Test your ideas on interesting (but small-ish) data sets of your choosing and/or design.

4. Add an SR1-based Trust Region optimization method to your code. Compare performance with various line search methods.