May 30, 2014

Tikhonov regularization

- $R(u) = \int_\Omega |u|^2 \, dx = \|u\|^2$

  i.e.

$$\min_u \quad \alpha\|u\|^2 + \|u - u^0\|^2$$

- Euler-Lagrange equation:

$$\begin{aligned} \alpha u + (u - u^0) &= 0 \\ (\alpha + 1)u &= u^0 \\ u &= \tfrac{1}{\alpha+1} u^0 \end{aligned}$$

- $\alpha \approx 0$ -> $u \approx u^0$
- $\alpha \approx \infty$ -> $u \approx 0$

Laplacian regularization

- $R(u) = \int_\Omega |\nabla u|^2 \, dx$

  i.e.

$$\min_u \quad \alpha\|\nabla u\|^2 + \|u - u^0\|^2$$

  The idea is to have small slopes, not small pixel values.

  For noisy images, slopes are large.

- Euler-Lagrange equation:

$$
\begin{aligned}
-\alpha\,\Delta u + (u - u^0) &= 0 \\
-\alpha\,\Delta u + u &= u^0
\end{aligned}
$$

- Finite difference approximation:

$$\frac{\alpha}{h^2}(4u_{i,j} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1}) + u_{ij} = u^0_{i,j}$$

Matrix form:

$$
\begin{aligned}
\alpha A u + u &= u^0 \\
(\alpha A + I) u &= u^0
\end{aligned}
$$

- If the solution u is still too noisy, repeat the procedure:

for k = 0, 1, . . . , K

Solve $(\alpha\ A + I)\ u^{k+1} = u^k$

end

Drawback: it tends to smear edges

Total variation regularization

- $R(u) = \int_{\Omega} |\nabla u|\, dx$

  i.e. $\min_{u} \quad \alpha \int_{\Omega} |\nabla u| dx + \|u - u^0\|^2$

  Idea: still min the slopes, but don't get punished too much by it.

- Euler-Lagrange equation:

$$
\begin{aligned}
-\alpha \nabla \cdot \left(\tfrac{1}{|\nabla u|}\right) \nabla u + (u - u^0) &= 0 \\
-\alpha \nabla \cdot \left(\tfrac{1}{|\nabla u|}\right) \nabla u + u &= u^0
\end{aligned}
$$

Note: in previous approach, the term 1/|grad u| is replaced by 1.

Near edges: $|\nabla u_{i,j}|$ is large => $\frac{1}{|\nabla u_{i,j}|}$ is small

=> $-\alpha \nabla \cdot \left(\frac{1}{|\nabla u_{i,j}|}\right) \nabla u \approx -\frac{\alpha}{|\nabla u_{i,j}|} \Delta u_{i,j} \approx 0$

=> $u_{i,j} \approx u_{i,j}^0$

On flat surfaces: $|\nabla u_{i,j}| \approx 0$ => $\frac{1}{|\nabla u_{i,j}|}$ is large

$\Rightarrow \quad -C \Delta u_{i,j} + u_{i,j} = u_{i,j}^0 \qquad (C = \text{large constant})$

$\Rightarrow$ more diffusion at (i,j)

$\Rightarrow$ $u_{i,j}$ is flat

- Euler-Lagrange is a nonlinear equation; the PDE coefficients depend on the solution.

- Finite difference approximation:

$$\begin{aligned} \alpha A(u) + u &= u^0 \\ (\alpha A(u) + I)u &= u^0 \end{aligned}$$

Matrix entries depend on the solution u.

- One solution method for solving nonlinear equations is fixed point iteration.

- Fix the coefficients to make the equation linear and then update the coefficients iteratively.

for k = 0, 1, . . . , K

Solve $(\alpha A(u^k) + I)\, u^{k+1} = u^k$

end

- In general, pick an initial guess, compute an approx. solution by a simple procedure, then repeat this process iteratively.

Iterative Methods

Splitting

Let $\quad A = M - N \quad\quad M \approx A$

Then
$$A\,x = b$$
$$(M - N)\,x = b$$
$$M\,x = N\,x + b$$

Define an iterative method by:

$$M\,x^{k+1} = N\,x^k + b$$

Then
$$\begin{aligned} x^{k+1} &= M^{-1} N\,x^k + M^{-1} b \\ &= M^{-1}(M - A)\,x^k + M^{-1} b \\ &= x^k + M^{-1}(b - A\,x^k) \end{aligned}$$

Note: If $M = A$, then
$$\begin{aligned} x^{k+1} &= x^k + A^{-1}(b - A\,x^k) \\ &= x^k + x - x^k = x \end{aligned}$$

-> one step convergence

But one needs to compute $A^{-1}(b - Ax^k)$

Goals:

(1) $M \approx A$

(2) M^{-1} is easy to compute

Richardson iteration

- M = 1/θ I   (θ is appropriately chosen)

  Then $M^{-1} = \theta I$

  Thus $x^{k+1} = x^k + \theta I (b - A x^k)$

  Consider the i-th entry of x^{k+1} :

$$x_i^{k+1} = x_i^k + \theta(b_i - \sum_{j=1}^{n} a_{i,j} x_j^k)$$

Algorithm

x^0 = initial guess

for k = 0, 1, 2, . . .

  for i = 1, 2, . . . , n

$$x_i^{k+1} = x_i^k + \theta(b_i - \sum_{j=1}^{n} a_{i,j} x_j^k)$$

  end

end

Note

Need 2 separate vectors x^k, x^{k+1}.

Jacobi iteration

- M = D = diagonal of A = $\begin{bmatrix} a_{1,1} & & \\ & \ddots & \\ & & a_{n,n} \end{bmatrix}$

Then

$$M^{-1} = \begin{bmatrix} a_{1,1}^{-1} & & \\ & \ddots & \\ & & a_{n,n}^{-1} \end{bmatrix}$$

Thus

$$\begin{aligned} x^{k+1} &= x^k + D^{-1}(b - Ax^k) \\ x_i^{k+1} &= x_i^k + \tfrac{1}{a_{i,i}}(b_i - \sum_{j=1}^{n} a_{i,j}x_j^k) \\ &= x_i^k + \tfrac{1}{a_{i,i}}(b_i - \sum_{j \neq i} a_{i,j}x_j^k - a_{i,i}x_i^k) \\ &= \tfrac{1}{a_{i,i}}(b_i - \sum_{j \neq i} a_{i,j}x_j^k) \end{aligned}$$

Interpretation

Let $r^k = b - A\,x^k$ (residual vector of x^k)

Then $x^k = x \iff r^k = 0$

Thus $||r^k||_2 \approx 0 \;\Rightarrow\; x^k \approx x$

Consider $$r_i^k = b_i - \sum_{j=1}^{n} a_{i,j}x_j^k = b_i - \sum_{j \neq i} a_{i,j}x_j^k - a_{i,i}x_i^k$$

In general, $r_i^k \neq 0$

Now modifying x_i^k so that $r_i^k = 0$.

i.e.

$$b_i - \sum_{j \neq i} a_{i,j} x_j^k - a_{i,i} x_i^{k+1} = 0$$

$$x_i^{k+1} = \frac{1}{a_{i,i}} (b_i - \sum_{j \neq i} a_{i,j} x_j^k)$$

-> Jacobi iteration

<u>Algorithm</u>

x^0 = initial guess

for k = 0, 1, 2, . . .

for i = 1, 2, . . . , n

$$x_i^{k+1} = \frac{1}{a_{i,i}} (b_i - \sum_{j \neq i} a_{i,j} x_j^k)$$

end

end

<u>Note</u>: need separate storage for x^k, x^{k+1}.

Gauss-Seidel iteration

Let $A = D - L - U$, where D = diagonal of A, L = strictly lower Δ part, U = strictly upper Δ part

i.e.

$$A = \begin{bmatrix} \ddots & & -U \\ & D & \\ -L & & \ddots \end{bmatrix}$$

Then GS iteration: $M = D - L$ = lower Δ part of A.

i.e. $$x^{k+1} = x^k + (D-L)^{-1}(b - A x^k)$$

Interpretation

Modify x_i^k so that $r_i^k = 0$. Use the new x_j^{k+1}, $j < i$, from the previous updates.

i.e. $$r_i^k = b_i - \sum_{j<i} a_{i,j} x_j^{k+1} - a_{i,i} x_i^{k+1} - \sum_{j>i} a_{i,j} x_j^k = 0$$

Thus $$x_i^{k+1} = \frac{1}{a_{i,i}}\left(b_i - \sum_{j<i} a_{i,j} x_j^{k+1} - \sum_{j>i} a_{i,j} x_j^k\right)$$

Algorithm

x^0 = initial guess

for $k = 0, 1, 2, \ldots$

for $i = 1, 2, \ldots, n$

$$x_i^{k+1} = \frac{1}{a_{i,i}}\left(b_i - \sum_{j<i} a_{i,j} x_j^{k+1} - \sum_{j>i} a_{i,j} x_j^k\right)$$

end

end

Note: No extra storage for x^{k+1}. x^{k+1} can be overwritten immediately.

Backward GS

- $M = D - U$: $x^{k+1} = x^k + (D-U)^{-1} (b - A x^k)$

Symmetric GS

A forward sweep followed by a backward sweep:

$$\begin{cases} x^{k+1/2} = x^k + (D-L)^{-1}(b - Ax^k) \\ x^{k+1} = x^{k+1/2} + (D-U)^{-1}(b - Ax^{k+1/2}) \end{cases}$$

$$\Leftrightarrow \qquad x^{k+1} = x^k + (D-U)^{-1} D (D-L)^{-1}(b - Ax^k)$$