ECS231

Least-squares problems

(Introduction to Randomized Algorithms)

May 21, 2019

Outline

1. linear least squares – review
2. Solving LS by sampling
3. Solving LS by randomized preconditioning
4. Gradient-based optimization – review
5. Solving LS by gradient-descent
6. Solving LS by stochastic gradient-descent

Review: Linear least squares

- Linear least squares problem

$$\min_x \|Ax - b\|_2$$

- Normal equation

$$A^T A x = A^T b$$

- Optimal solution

$$x = A^+ b$$

Solving LS by sampling

- MATLAB demo code: `lsbysampling.m`

```
>> ...
>> A = rand(m,n); b = rand(m,1);
>> sampled_rows = find( rand(m,1) < 10*n*log(n)/m );
>> A1 = A(sampled_rows,:);
>> b1 = b(sampled_rows);
>> x1 = A1\b1;
>> ...
```

- *Further reading: Avron et al, SIAM J. Sci. Comput., 32:1217-1236, 2010*

Solving LS by randomized preconditioning

- Linear least squares problem

$$\min_x \|A^T x - b\|_2$$

- Normal equation

$$(AA^T)x = Ab$$

- If we can find a P such that $P^{-1}A$ is well-conditioned, then it yields

$$\begin{aligned} x &= (AA^T)^{-1}Ab \\ &= P^{-T} \cdot (P^{-1}A \cdot (P^{-1}A)^T)^{-1} \cdot P^{-1}A \cdot b \end{aligned}$$

Solving LS by randomized preconditioning

- MATLAB demo code: `lsbyrandprecond.m`

```
>> ...
>> ell = m+4;
>> G = randn(n,ell);
>> S = A*G;      % sketching of A
>> [Q,R,E]=qr(S');  % QR w. col. pivoting S'*E = Q*R
>> P = E*R(1:m,1:m)';  % preconditioner P
>> B = P\A;
>> PAcondnum  = cond(B) % the condition number
>> ...
```

- *Further reading: Coakley et al, SIAM J. Sci. Comput., 33:849-868, 2011*

Review: Gradient-based optimization

- Optimization problem

$$x^* = \operatorname*{argmin}_x f(x)$$

- Gradient: $\nabla_x f(x)$
  The first-order approximation

$$f(x + \Delta x) = f(x) + \Delta x^T \nabla_x f(x) + O(\|\Delta x\|_2^2)$$

  Directional derivative: $\frac{\partial}{\partial \alpha} f(x + \alpha u) = u^T \nabla_x f(x)$
- To min $f(x)$, we would like to find the direction u in which f decreases the fastest. Using the directional derivative,

$$f(x + \alpha u) = f(x) + \alpha u^T \nabla_x f(x) + O(\alpha^2)$$

  Note that

$$\begin{aligned} \min_{u, u^T u = 1} u^T \nabla_x f(x) &= \min_{u, u^T u = 1} \|u\|_2 \|\nabla_x f(x)\|_2 \cos\theta \\ &= -\|\nabla_x f(x)\|_2 \end{aligned}$$

  when u is the opposite of $\nabla_x f(x)$. Therefore, the steepest descent direction $u = -\nabla_x f(x)$.

Review: Gradient-based optimization, cont'd

- The method of steepest descent

$$x' = x - \epsilon \cdot \nabla_x f(x),$$

where the "learning rate" ϵ can be chosen as follows:

1. $\epsilon =$ small const.
2. $\min_\epsilon f(x - \epsilon \cdot \nabla_x f(x))$
3. evaluate $f(x - \epsilon \nabla_x f(x))$ for several different values of ϵ and choose the one that results in the smallest objective function value.

Solving LS by gradient-descent

- Minimization problem

$$\min_x f(x) = \min_x \frac{1}{2}\|Ax - b\|_2^2$$

- Gradient: $\nabla_x f(x) = A^T Ax - A^T b$
- The method of gradient descent:
  - set the stepsize ϵ and tolerance δ to small positive numbers.
  - while $\|A^T Ax - A^T b\|_2 > \delta$ do

$$x \leftarrow x - \epsilon \cdot (A^T Ax - A^T b)$$

  - end while

Solving LS by gradient-descent

MATLAB demo code: `lsbygd.m`

```
>> ...
>> r = A'*(A*x - b);
>> xp = x - tau*r;
>> res(k) = norm(r);
>> if res(k) <= tol, ... end
>> ...
>> x = xp;
>> ...
```

Solve LS by stochastic gradient descent

- Minimization problem:

$$x_* = \underset{x}{\operatorname{argmin}} \frac{1}{2}\|Ax - b\|_2^2 = \underset{x}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n} f_i(x) = \underset{x}{\operatorname{argmin}} \mathbb{E} f_i(x)$$

  where $f_i(x) = \frac{n}{2}(\langle a_i, x\rangle - b_i)^2$ and $a_1, a_2 ...$ are the rows of A.
- Gradient: $\nabla_x f_i(x) = n(\langle a_i, x\rangle - b_i)a_i$.
- The stochastic gradient descent (SGD) method solves the LS problem by iterative moving in the gradient direction of a selected function f_{i_k}:

$$x_{k+1} \leftarrow x_k - \gamma \cdot \nabla f_{i_k}(x_k)$$

  where index i_k is selected *randomly* in the kth iteration:
  - uniformally at random, or
  - weighted sampling [1]

[1] D. Needell et al, Stochastic gradient descent, weighted sampling, and the randomized Kaczmarz algorithm, Math. Program. Ser. A (2016) 155:549-573.

Solve LS by stochastic gradient descent

MATLAB demo code: `lsbysgd.m`

```
>> ...
>> s = rand;
>> i = sum(s >= cumsum([0, prob]));  % with probability prob(i)
>> dx = n*(A(i,:)*x0 - b(i))*A(i,:);
>> x = x0 - (gamma/(n*prob(i)))*dx'; % weighted SGD
>> ...
```