03 - Solving the least squares problem in practice

Numerical Methods for Deep Learning

January 15, 2018

Iterative solvers for least-squares regression

Last time we solved

$$\min_{\mathbf{W}} \|\mathbf{Y}\mathbf{W} - \mathbf{C}\|^2$$

directly using $\mathbf{W} = (\mathbf{Y}^\top \mathbf{Y})\mathbf{Y}^\top \mathbf{C}$.

Problems: generating $\mathbf{Y}^\top \mathbf{Y}$ and solving normal equations is too costly for large-scale problems.

Iterative solvers for least-squares regression

Last time we solved

$$\min_{\mathbf{W}} \|\mathbf{Y}\mathbf{W} - \mathbf{C}\|^2$$

directly using $\mathbf{W} = (\mathbf{Y}^\top\mathbf{Y})\mathbf{Y}^\top\mathbf{C}$.

Problems: generating $\mathbf{Y}^\top\mathbf{Y}$ and solving normal equations is too costly for large-scale problems.

Today: Iterative methods that avoid working with $\mathbf{Y}^\top\mathbf{Y}$

- Steepest decent
- Conjugate gradient

Iterative methods

General idea - obtain a sequence $\mathbf{W}_1, \ldots, \mathbf{W}_j, \ldots$ that converges to least-squares solution $\mathbf{W}^*$

$$\mathbf{W}_j \longrightarrow \mathbf{W}^*, \quad \text{for} \quad j \to \infty.$$

How fast does the sequence converge? Assume

$$\|\mathbf{W}_{j+1} - \mathbf{W}^*\| < \gamma_j \|\mathbf{W}_j - \mathbf{W}^*\|$$

where all $\gamma_j < 1$. Then

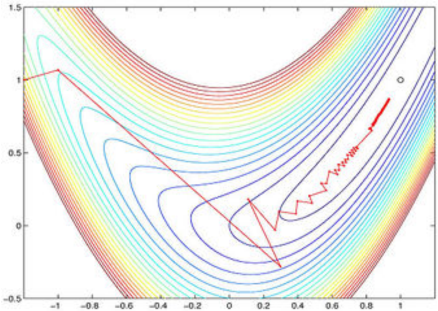
- If γ_j is bounded away from 0 and 1 the convergence is linear
- If $\gamma_j \to 0$ the convergence is superlinear
- If $\gamma_j \to 1$ the convergence is sublinear

The sequence converges quadratically if

$$\|\mathbf{W}_{j+1} - \mathbf{W}^*\| < \gamma_j \|\mathbf{W}_j - \mathbf{W}^*\|^2$$

Steepest descent

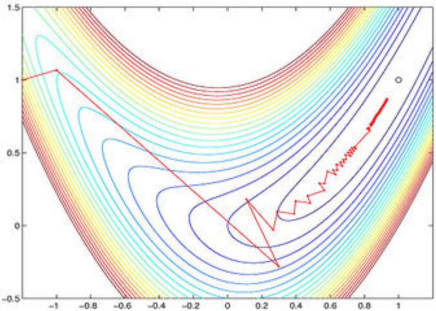
Most basic iterative technique for solving $\min_{\mathbf{x}} f(\mathbf{x})$



$$\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{s}_j \quad \text{with} \quad \mathbf{s}_j = -\frac{\nabla f(\mathbf{x}_j)}{\|\nabla f(\mathbf{x}_j)\|}.$$

Steepest descent

Most basic iterative technique for solving $\min_{\mathbf{x}} f(\mathbf{x})$



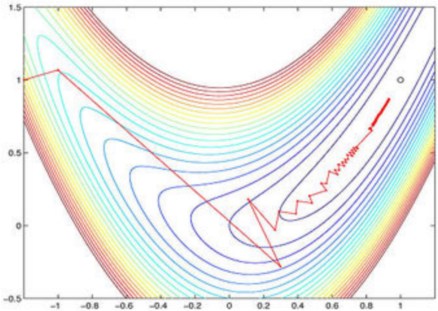
$$\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{s}_j \quad \text{with} \quad \mathbf{s}_j = -\frac{\nabla f(\mathbf{x}_j)}{\|\nabla f(\mathbf{x}_j)\|}.$$

Interpretation 1: $\mathbf{s}_{j+1}$ maximizes local descent, i.e., solves

$$\min_s f(\mathbf{x}_j) + \mathbf{s}^\top \nabla f(\mathbf{x}_j) \quad \text{subject to} \quad \|\mathbf{s}\|_2 = 1.$$

Steepest descent

Most basic iterative technique for solving $\min_{\mathbf{x}} f(\mathbf{x})$



$$\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{s}_j \quad \text{with} \quad \mathbf{s}_j = -\frac{\nabla f(\mathbf{x}_j)}{\|\nabla f(\mathbf{x}_j)\|}.$$

Interpretation 1: $\mathbf{s}_{j+1}$ maximizes local descent, i.e., solves

$$\min_s f(\mathbf{x}_j) + \mathbf{s}^\top \nabla f(\mathbf{x}_j) \quad \text{subject to} \quad \|\mathbf{s}\|_2 = 1.$$

Interpretation 2: $\mathbf{s}_j$ is orthogonal to level sets of f at $\mathbf{x}_j$.

Steepest descent for least-squares

Consider now

$$f(\mathbf{w}) = \frac{1}{2}\|\mathbf{Y}\mathbf{w} - \mathbf{c}\|^2 \quad \text{with} \quad \nabla_{\mathbf{w}} f(\mathbf{w}) = \mathbf{Y}^\top(\mathbf{Y}\mathbf{w} - \mathbf{c}).$$

Steepest descent direction is $\mathbf{s}_j = \mathbf{Y}^\top(\mathbf{c} - \mathbf{Y}\mathbf{w}_j)$ and

$$\mathbf{w}_{j+1} = \mathbf{w}_j + \alpha_j \mathbf{s}_j$$

How to choose α_j?

Steepest descent for least-squares

Consider now

$$f(\mathbf{w}) = \frac{1}{2}\|\mathbf{Y}\mathbf{w} - \mathbf{c}\|^2 \quad \text{with} \quad \nabla_{\mathbf{w}} f(\mathbf{w}) = \mathbf{Y}^\top(\mathbf{Y}\mathbf{w} - \mathbf{c}).$$

Steepest descent direction is $\mathbf{s}_j = \mathbf{Y}^\top(\mathbf{c} - \mathbf{Y}\mathbf{w}_j)$ and

$$\mathbf{w}_{j+1} = \mathbf{w}_j + \alpha_j \mathbf{s}_j$$

How to choose α_j? Idea: Minimize f along direction $\mathbf{s}_j$

$$\alpha_j = \arg\min_{\alpha} f(\mathbf{w}_j + \alpha \mathbf{s}_j) = \arg\min_{\alpha} \frac{1}{2}\|\alpha \mathbf{Y}\mathbf{s}_j - \mathbf{r}_j\|^2$$

with residual $\mathbf{r}_j = \mathbf{c} - \mathbf{Y}\mathbf{w}_j$.

Steepest descent for least-squares

Consider now

$$f(\mathbf{w}) = \frac{1}{2}\|\mathbf{Y}\mathbf{w} - \mathbf{c}\|^2 \quad \text{with} \quad \nabla_{\mathbf{w}} f(\mathbf{w}) = \mathbf{Y}^\top(\mathbf{Y}\mathbf{w} - \mathbf{c}).$$

Steepest descent direction is $\mathbf{s}_j = \mathbf{Y}^\top(\mathbf{c} - \mathbf{Y}\mathbf{w}_j)$ and

$$\mathbf{w}_{j+1} = \mathbf{w}_j + \alpha_j \mathbf{s}_j$$

How to choose α_j? Idea: Minimize f along direction $\mathbf{s}_j$

$$\alpha_j = \arg\min_\alpha f(\mathbf{w}_j + \alpha \mathbf{s}_j) = \arg\min_\alpha \frac{1}{2}\|\alpha \mathbf{Y}\mathbf{s}_j - \mathbf{r}_j\|^2$$

with residual $\mathbf{r}_j = \mathbf{c} - \mathbf{Y}\mathbf{w}_j$.

This leads to simple quadratic equation in 1D whose solution is

$$\alpha_j = \frac{\mathbf{r}_j^\top \mathbf{Y}\mathbf{s}_j}{\|\mathbf{Y}\mathbf{s}_j\|^2}$$

Algorithm: Steepest descent for least-squares

for $j = 1, \ldots$

- Compute residual $\mathbf{r}_j = \mathbf{c} - \mathbf{Y}\mathbf{w}_j$
- Compute the SD direction $\mathbf{s}_j = \mathbf{Y}^\top \mathbf{r}_j$
- Compute step size $\alpha_j = \frac{\mathbf{r}_j^\top \mathbf{Y}\mathbf{s}_j}{\|\mathbf{Y}\mathbf{s}_j\|^2}$
- Take the step $\mathbf{w}_{j+1} = \mathbf{w}_j + \alpha_j \mathbf{s}_j$

Algorithm: Steepest descent for least-squares

for $j = 1, \ldots$

- Compute residual $\mathbf{r}_j = \mathbf{c} - \mathbf{Y}\mathbf{w}_j$
- Compute the SD direction $\mathbf{s}_j = \mathbf{Y}^\top \mathbf{r}_j$
- Compute step size $\alpha_j = \frac{\mathbf{r}_j^\top \mathbf{Y}\mathbf{s}_j}{\|\mathbf{Y}\mathbf{s}_j\|^2}$
- Take the step $\mathbf{w}_{j+1} = \mathbf{w}_j + \alpha_j \mathbf{s}_j$

Converges linearly, i.e.,

$$\|\mathbf{W}_{j+1} - \mathbf{W}^*\| < \gamma \|\mathbf{W}_j - \mathbf{W}^*\| \quad \text{with} \quad \gamma \approx \left| \frac{\kappa - 1}{\kappa + 1} \right|$$

Here, κ depends on condition number of $\mathbf{Y}$, i.e.,

$$\kappa \approx \frac{\sigma^2_{\mathbf{min}}}{\sigma^2_{\mathbf{max}}}$$

Can be painfully slow for ill-conditioned problems

Accelerating steepest descent: Post-conditioning

Idea: Improve convergence by transforming the problem

$$f(\mathbf{w}) = \frac{1}{2}\|\mathbf{YSS}^{-1}\mathbf{w} - \mathbf{c}\|^2$$

Here: $\mathbf{S}$ is invertible
Solve in two steps:

1. Set $\mathbf{z} = \mathbf{S}^{-1}\mathbf{w}$ and compute

$$\mathbf{z}^* \arg\min_{\mathbf{z}} \frac{1}{2}\|\mathbf{YSz} - \mathbf{c}\|^2$$

2. Then $\mathbf{w} = \mathbf{Sz}$.

Pick $\mathbf{S}$ such that $\mathbf{YS}$ is better conditioned.

Accelerating steepest descent: Post-conditioning

Idea: Improve convergence by transforming the problem

$$f(\mathbf{w}) = \frac{1}{2}\|\mathbf{YSS}^{-1}\mathbf{w} - \mathbf{c}\|^2$$

Here: $\mathbf{S}$ is invertible
Solve in two steps:

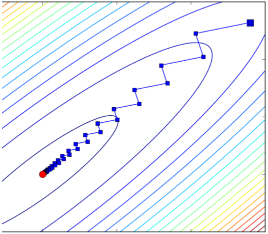
1. Set $\mathbf{z} = \mathbf{S}^{-1}\mathbf{w}$ and compute

$$\mathbf{z}^* \arg\min_{\mathbf{z}} \frac{1}{2}\|\mathbf{YSz} - \mathbf{c}\|^2$$

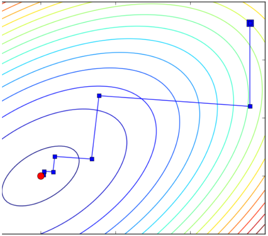
2. Then $\mathbf{w} = \mathbf{Sz}$.

Pick $\mathbf{S}$ such that $\mathbf{YS}$ is better conditioned.

original problem:



post-conditioned:



Exercise: Steepest descent for least-squares

Goal: Program steepest descent and solve some simple problem.

To verify your code generate data using

$$\mathbf{c} = \mathbf{Y}\mathbf{w}_{\mathrm{true}} + \boldsymbol{\epsilon}.$$

where $\boldsymbol{\epsilon}$ is random with zero mean and standard deviation 0.1 and

$$\mathbf{Y} = \begin{pmatrix} 1 & 1+a \\ 1 & 1+2a \\ 1 & 1+3a \end{pmatrix} \quad \text{and} \quad \mathbf{w}_{\mathrm{true}} = \begin{pmatrix} 1 \\ 1.2 \end{pmatrix}.$$

Plot errors $\|\mathbf{w}_j - \mathbf{w}^*\|$ for $j = 1, \ldots$ and $a \in \{1, 10^{-2}, 10^{-5}\}$.

Conjugate gradient method for least-squares

CG is designed to solve quadratic optimization problems

$$\min_{\mathbf{w}} \frac{1}{2}\mathbf{w}^\top \mathbf{H}\mathbf{w} - \mathbf{b}^\top \mathbf{w}$$

with $\mathbf{H}$ symmetric positive definite. In our case

$$\arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{Y}\mathbf{w} - \mathbf{c}\|^2 = \arg\min_{\mathbf{w}} \frac{1}{2}\mathbf{w}^\top \mathbf{Y}^\top \mathbf{Y}\mathbf{w} - \mathbf{w}^\top \mathbf{Y}^\top \mathbf{c}$$

Conjugate gradient improves over SD by using previous steps (not a memory-less method) and constructing a basis for the solution.

Facts:

- terminates after at most n steps (in exact arithmetic)
- good solutions for $j \ll n$
- convergence $\gamma_j \approx \left|\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right|^j$

CGLS: Conjugate Gradient Least Squares

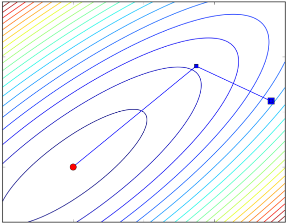
```
function w = cgls(Y,c,k)
n = size(Y,2);
w = zeros(n,1);
d = Y'*c;   r = c;
normr2 = d'*d;
for j=1:k
    Ad = Y*d; alpha = normr2/(Ad'*Ad);
    w  = w + alpha*d;
    r  = r - alpha*Ad;
    s  = Y'*r;
    normr2New = s'*s;
    beta = normr2New/normr2;
    normr2 = normr2New;
    d = s + beta*d;
end
```

Conjugate Gradient Least-Squares

- Uses the structure of the problem to obtain stable implementation
- Typically converges much faster than SD
- Accelerate using post conditioning

$$\min \frac{1}{2}\|\mathbf{YSS}^{-1}\mathbf{w} - \mathbf{c}\|^2$$

- Faster convergence when eigenvalues of $\mathbf{S}^\top\mathbf{Y}^\top\mathbf{YS}$ are clustered.



Iterative regularization

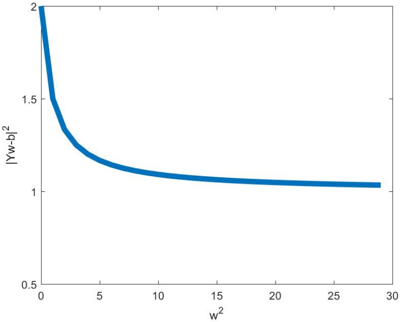
- Assume that $\mathbf{Y}$ has a null space
- The matrix $\mathbf{Y}^\top\mathbf{Y}$ is not invertible
- Can we still use CGLS to solve(?) the least squares problem

What are the properties of CGLS iterations?

Iterative regularization: L-Curve

The CGLS algorithm has the following properties

- For each iteration $\|\mathbf{Y}\mathbf{w}_j - \mathbf{c}\|^2 \leq \|\mathbf{Y}\mathbf{w}_{j-1} - \mathbf{c}\|^2$
- If starting from $\mathbf{w} = 0$ then $\|\mathbf{w}_j\|^2 \geq \|\mathbf{w}_{j-1}\|^2$
- $\mathbf{w}_1, \mathbf{w}_2, \ldots$ converges to the minimum norm solution of the problem
- Plotting $\|\mathbf{w}_j\|^2$ vs $\|\mathbf{Y}\mathbf{w}_j - \mathbf{c}\|^2$ typically has the shape of an L-curve



Cross Validation

Finding good least-squares solution requires good parameter selection.

- α is using Tikhonov (weight decay)
- number of iteration (for SD and CGLS)

Suppose that we have two different "solutions"

$$\begin{array}{lcll} \mathbf{w}_1 & \rightarrow & \|\mathbf{w}_1\|^2 = \eta_1 & \|\mathbf{Y}\mathbf{w}_1 - \mathbf{c}\|^2 = \rho_1. \\ \mathbf{w}_2 & \rightarrow & \|\mathbf{w}_2\|^2 = \eta_2 & \|\mathbf{Y}\mathbf{w}_2 - \mathbf{c}\|^2 = \rho_2. \end{array}$$

How to decide which one is better?

Cross Validation

Measure how well can each of the solutions predict new data.

Let $\{\mathbf{Y}_{\mathrm{CV}}, \mathbf{c}_{\mathrm{CV}}\}$ be data that is **not used** for the training

Then if $\|\mathbf{Y}_{\mathrm{CV}}\mathbf{w}_1 - \mathbf{c}_{\mathrm{CV}}\|^2 \leq \|\mathbf{Y}_{\mathrm{CV}}\mathbf{w}_2 - \mathbf{c}_{\mathrm{CV}}\|^2$ then $\mathbf{w}_1$ is a better solution that $\mathbf{w}_2$.

Cross Validation

Measure how well can each of the solutions predict new data.

Let $\{\mathbf{Y}_{\mathrm{CV}}, \mathbf{c}_{\mathrm{CV}}\}$ be data that is **not used** for the training

Then if $\|\mathbf{Y}_{\mathrm{CV}}\mathbf{w}_1 - \mathbf{c}_{\mathrm{CV}}\|^2 \leq \|\mathbf{Y}_{\mathrm{CV}}\mathbf{w}_2 - \mathbf{c}_{\mathrm{CV}}\|^2$ then $\mathbf{w}_1$ is a better solution that $\mathbf{w}_2$.

In general, if the solution depends on some hyper-parameter θ then the best one is

$$\theta^* = \arg\min \|\mathbf{Y}_{\mathrm{CV}}\mathbf{w}(\theta) - \mathbf{c}_{\mathrm{CV}}\|^2.$$

Cross Validation

To assess the final quality of the solution cross validation is not sufficient (why?).

Need a final testing set.

Procedure

- Divide the data into 3 groups $\{\mathbf{Y}_{\mathrm{train}}, \mathbf{Y}_{\mathrm{CV}}, \mathbf{Y}_{\mathrm{test}}\}$.
- Use $\mathbf{Y}_{\mathrm{train}}$ to estimate $\mathbf{w}(\theta)$
- Use $\mathbf{Y}_{\mathrm{CV}}$ to estimate θ
- Use $\mathbf{Y}_{\mathrm{test}}$ to assess the quality of the solution

Cross Validation

To assess the final quality of the solution cross validation is not sufficient (why?).

Need a final testing set.

Procedure

- Divide the data into 3 groups $\{\mathbf{Y}_{\text{train}}, \mathbf{Y}_{\text{CV}}, \mathbf{Y}_{\text{test}}\}$.
- Use $\mathbf{Y}_{\text{train}}$ to estimate $\mathbf{w}(\theta)$
- Use $\mathbf{Y}_{\text{CV}}$ to estimate θ
- Use $\mathbf{Y}_{\text{test}}$ to assess the quality of the solution

Important - we are not allowed to use $\mathbf{Y}_{\text{test}}$ to tune parameters!

Coding: Iterative Methods for Regression

Outline:

- Dataset: MNIST / CIFAR 10
- write a steepest descent specific to the problem

  `function x = sdLeastSquares(A,b,x0,maxIter)`
- write a conjugate gradient code

  `function x = cgLeastSquares(A,b,maxIter)`