LEUPHANA UNIVERSITY
MACHINE LEARNING SEMINAR
WINTER 22/23

INTRODUCTION TO CONJUGATE GRADIENT METHOD

Corelleta Kaseke
Huyen Priet-Nguyen
Osman Deger

AGENDA

01
INTRODUCTION
&
RECALL SD

02
THE ALGORITHM
&
EXAMPLE

03
MINI COMPARISON
of
SD & CG

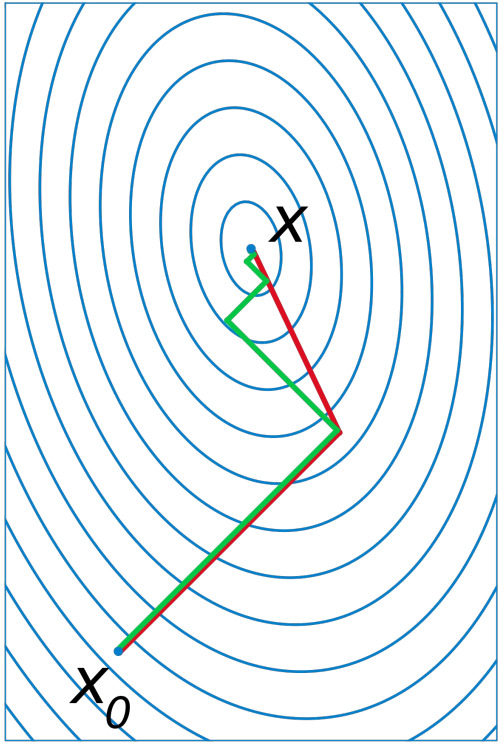
04
THE LINEAR CASE
&
EXTENSIONS

05
NON LINEAR CASE
&
EXAMPLE

06
OVERALL ANALYSIS
&
CONCLUSION

INTRODUCTION

- Which would you prefer?
  - Green or Red
- Conjugate Gradient Method (CGM) in a nutshell
  - Hestenes and Stiefel (1952)
  - Linear, N Equations, N unknowns $Ax = b$
  - Symmetric & Positive Definite (SPD), Sparse
- Fast Convergence, but what else?
  - Cost Efficiency, Flexibility



Drawing by Oleg Alexandrov on wikimedia commons

Recap of the Steepest Descent (SD)

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c \quad (1)$$ Quadratic Equation to be solved

$$Ax = b \quad (2)$$ The minimization rule

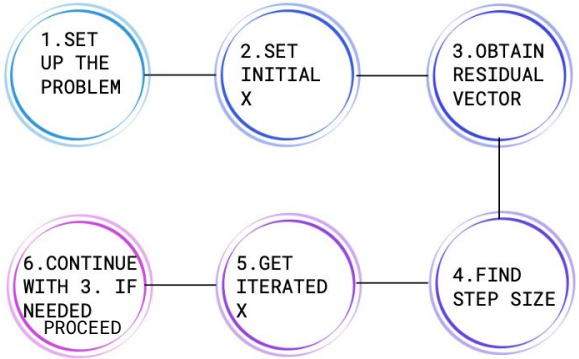
$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} r_{(i)} \quad (3)$$ The iteration equation

$$r_{(i)} = b - Ax_{(i)} \quad (4)$$ The residual

$$-f'(x_{(i)}) = b - Ax_{(i)} \quad (5)$$ The gradient

$$\frac{d}{d\alpha} f(x_{(i+1)}) = f'(x_{(i+1)})^T \frac{d}{d\alpha} f(x_{(i+1)}) = f'(x_{(i+1)})^T r_{(i)} = -r_{(i+1)}^T r_{(i)} = 0 \quad (6)$$ Finding the step size α

$$\alpha_{(i)} = \frac{r_{(i)}^T r_{(i)}}{r_{(i)}^T A r_{(i)}} \quad (7)$$ The step size

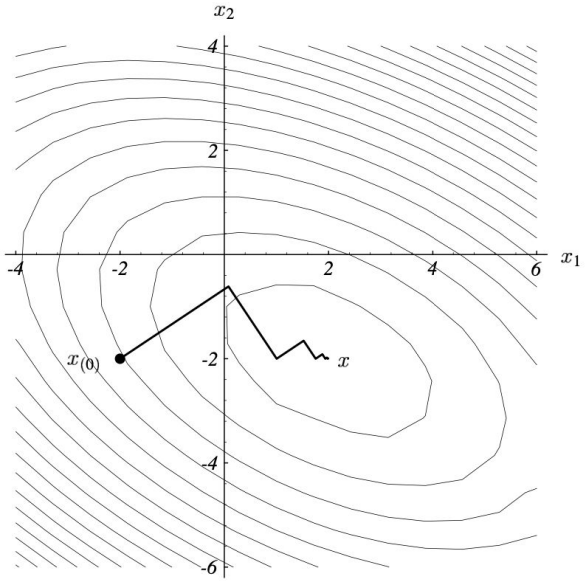


Shewchuk (1994)

THE MATH BEHIND- Numerical Example on SD

$$A = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, b = \begin{bmatrix} 2 \\ -8 \end{bmatrix}, x_{(0)} = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$$

	0	1	…
$r_{(i)} = b - Ax_{(i)}$	$r_{(0)} = \begin{bmatrix} 12 \\ 8 \end{bmatrix}$	$r_{(1)} = \begin{bmatrix} 2.988 \\ -4.476 \end{bmatrix}$	
$\alpha_{(i)} = \frac{r_{(i)}^T r_{(i)}}{r_{(i)}^T A r_{(i)}}$	$\alpha_{(0)} = 208/1200$	$\alpha_{(1)} = 28.96/93.49$	
$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)}$	$x_{(1)} = \begin{bmatrix} 0.08 \\ -0.614 \end{bmatrix}$	$x_{(2)} = \begin{bmatrix} 1.01 \\ -2.004 \end{bmatrix}$	



Shewchuk (1994)

THE ALGORITHM - CONJUGATE GRADIENT METHOD

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)} \quad (8)$$ Update rule of x

$$d_{(i)}^T A d_{(j)} = 0 \quad (9)$$ A-conjugates

$$r_{(i+1)} = b - Ax_{(i)} = r_{(i)} - \alpha_{(i)} A d_{(i)} \quad (10)$$ Residual

$$\alpha_{(i)} = \frac{d_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}} \quad (11)$$ The step-size

$$\beta_{(i+1)} = \frac{r_{(i+1)}^T r_{(i+1)}}{r_{(i)}^T r_{(i)}} \quad (12)$$ Parameter β

$$d_{(i+1)} = r_{(i+1)} + \beta_{(i+1)} d_{(i)} \quad (13)$$ Update rule of d

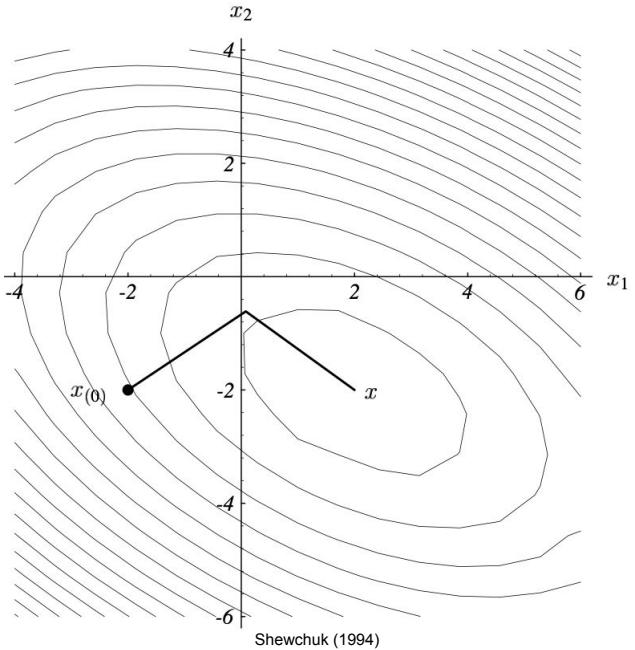
1. Problem Setup
2. Start with initial X
3. Find residual(direction)
4. Find step size
5. Find next X vector
6. Find next residual
7. Calculate β
8. Find the conjugate direction
9. Repeat until convergence

Shewchuk (1994)

THE ALGORITHM - NUMERICAL EXAMPLE ON CGM

$$A = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, b = \begin{bmatrix} 2 \\ -8 \end{bmatrix}, x_{(0)} = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$$

	0	1
$r_{(i)} = b - Ax_{(i)}$	$r_{(0)} = \begin{bmatrix} 12 \\ 8 \end{bmatrix}$	$r_{(1)} = \begin{bmatrix} 2.988 \\ -4.476 \end{bmatrix}$
$\alpha = \frac{d_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}}$	$\alpha = 208/1200$	$\alpha = 28.98/70.355$
$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)}$	$x_{(1)} = \begin{bmatrix} 0.08 \\ -0.614 \end{bmatrix}$	$x_{(2)} = \begin{bmatrix} 2 \\ -2 \end{bmatrix}$
$r_{(i)} = b - Ax_{(i)}$	$r_{(1)} = \begin{bmatrix} 2.988 \\ -4.476 \end{bmatrix}$	$r_{(2)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$
$\beta = \frac{r_{(i+1)}^T r_{(i+1)}}{r_{(i)}^T r_{(i)}}$	$\beta = 28.96/208$	
$d_{(i+1)} = r_{(i+1)} + \beta d_{(i)}$	$d_{(1)} = \begin{bmatrix} 4.658 \\ -3.366 \end{bmatrix}$	



Shewchuk (1994)

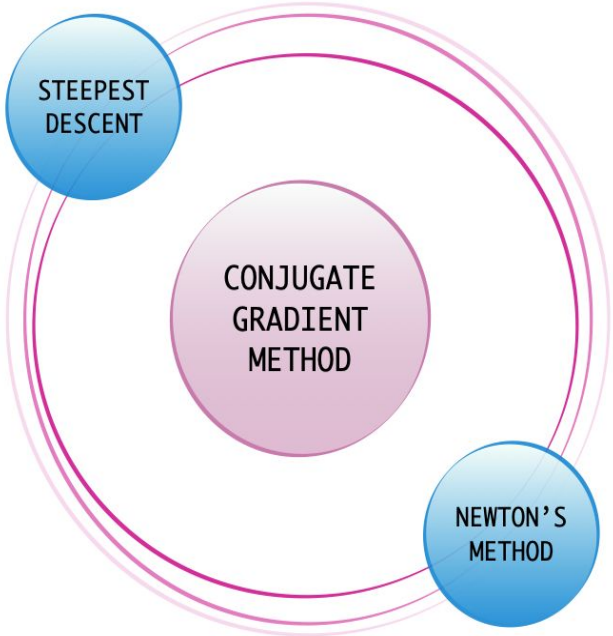
WHY NOT USE FASTER METHODS?

SD Advantages:

- Simple & Reliable
- Cheap Iterations
- Common Starting Directions

SD Disadvantages:

- Slower convergence
- Convergence rate depends adversely on the condition number of the Hessian (A)



NM Advantages:

- Fast convergence

NM Disadvantages:

- Requires calculation of Hessian inverse (memory and computation costs)
- For non-quadratics it can diverge or converge at saddle points

Deb(2014); Shewchuk (1994); Schraudolph and Graepel (2002)

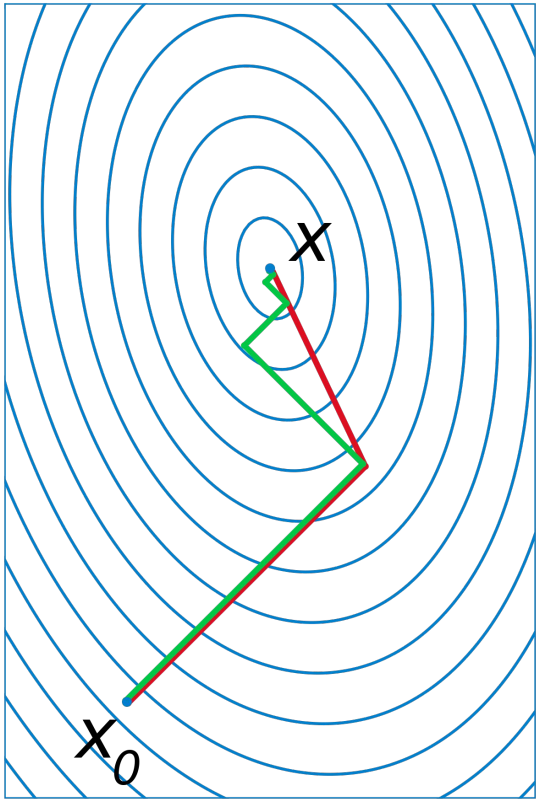
WHAT MAKES CGM COMPETITIVE?

Problems with SD

- Finds the steepest descent direction
- Steps orthogonal to the last step
- Retracing of steps

CG Propositions

- Finds the steepest descent and good direction
- New direction is A-orthogonal to the last direction
- Movement in a direction should be done once



Deb(2014); Hestenes and Stiefel (1952); Shewchuk (1994); Schraudolph and Graepel (2002); Drawing by Oleg Alexandrov on wikimedia commons

For non-SPD and non-square matrices

- Normal equation: If A is nonsingular, nonsymmetric

$$Ax = b \tag{14}$$

$$A^T Ax = A^T b \tag{15}$$

- Linear regression Problem: Minimize least square function $\min_x \|Ax - b\|^2$
  - Setting the derivative equal to zero leads to equation (15), where A is non-square.
  - $A^T A$ is SPD
  - If (14) is overconstrained, $A^T A$ is nonsingular

Problem: the normal equation (15) might converge slowly when space the dimension is large.

Hestenes and Stiefel (1952); Shewchuk (1994)

Preconditioned CGM to accelerate convergence

- A bound on convergence rate of CGM

$$\|e_{(i)}\|_A \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^i \|e_{(0)}\|_A, \tag{16}$$

$$\kappa = \left|\frac{\lambda_{\max}}{\lambda_{\min}}\right|, \quad \|e\|_A^2 = < e, Ae > . \tag{17}$$

- M approximates A, M is invertible and SPD

$$M^{-1}Ax = M^{-1}b, \quad \kappa(M^{-1}A) \ll \kappa(A) \tag{18}$$

- $M^{-1}A$ is not SPD generally → Some factorization $M = E^T E$
- Eigenvalues of $M^{-1}A$ and $E^{-1}AE^{-T}$ are equal $Ax = b$ is transformed to

$$E^{-1}AE^{-T}\hat{x} = E^{-1}b, \quad \hat{x} = E^T x \tag{19}$$

(19) can be applied to CGM

Shewchuk (1994)

LINEAR CGM - EXTENSIONS

Setting

$$\hat{r}_{(i)} = E^{-1} r_{(i)},\ \hat{d}_{(i)} = E^T d_{(i)} \tag{20}$$

CGM for (19) becomes the **untransformed Preconditioned CGM**

$$r_{(0)} = b - Ax_{(0)}, \tag{21}$$

$$d_{(0)} = M^{-1} r_{(0)}, \tag{22}$$

$$\alpha_{(i)} = \frac{r_{(i)}^T M^{-1} r_{(i)}}{d_{(i)}^T A d_{(i)}}, \tag{23}$$

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)}, \tag{24}$$

$$r_{(i+1)} = r_{(i)} - \alpha_{(i)} A d_{(i)}, \tag{25}$$

$$\beta_{(i+1)} = \frac{r_{(i+1)}^T M^{-1} r_{(i+1)}}{r^T{}_{(i)} M^{-1} r_{(i+1)} + \beta_{(i+1)} d_{(i)}}. \tag{26}$$

Shewchuk (1994)

Some generic preconditioners

- **Jacobi preconditioner:** $M = \mathrm{diag}(A)$

- **Incomplete Cholesky CGM:**
  - $M = LL^T$ is the Cholesky factorization, where L is lower triangular matrix.
  - $M^{-1}z = L^{T-}L^{-1}z$ via forward/backward substitution.
  - M approximates by ignoring small numbers of A

- **Trade-off** between enhanced convergence with the extra cost in preconditioned CGM

Shewchuk (1994)

Example of Jacobi Preconditioned CGM

$$A = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, b = \begin{bmatrix} 2 \\ -8 \end{bmatrix}, x_{(0)} = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$$

$$\text{M=} \begin{bmatrix} 3 & 0 \\ 0 & 6 \end{bmatrix}$$

$$\kappa(M^{-1}A) = 2.8,\ \kappa(A) = 3.5$$

	CGM	Jacobi Preconditioner
Residual at iteration 1 (L2 norm)	5.3817	3.9353
Residual at iteration 2 (L2 norm)	0.0000	0.0000

Shewchuk (1994)

Example of incomplete Cholesky Preconditioned CGM

$$A=\begin{bmatrix} 0.7444 & -0.5055 & -0.0851 \\ -0.5055 & 3.4858 & 0.0572 \\ -0.0851 & 0.0572 & 0.4738 \end{bmatrix} \quad M=\begin{bmatrix} 0.7444 & -0.5055 & 0 \\ -0.5055 & 3.4858 & 0 \\ 0 & 0 & 0.4738 \end{bmatrix}$$

$$\kappa(M^{-1}A) = 1.04,\ \kappa(A) = 8.01$$

$$b = [-0.0043 \quad 2.2501 \quad 0.2798]^T,$$

$$x_0 = [3 \quad 1 \quad -7]^T$$

	CGM	Incomplete Cholesky
Residual at iteration 1 (L2 norm)	3.1320	0.4381
Residual at iteration 2 (L2 norm)	0.4488	0.0051
Residual at iteration 3 (L2 norm)	0.0000	0.0000

Code modified for incomplete Cholesky from https://towardsdatascience.com/complete-step-by-step-conjugate-gradient-algorithm-from-scratch-202c07fb52a8

Changes in non-linear CGM

- Recursive for residual cannot be used
- More complicated learning rate
- Different choices of Gram-Schmidt coefficients
- Hessian is variant with respect to x

Outline of Fletcher-Reeves method

$$d_{(0)} = r_{(0)} = -f'(x_{(0)}) \tag{27}$$

$$\min_{\alpha_{(i)}} f(x_{(i)} + \alpha_{(i)} d_{(i)}) \tag{28}$$

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)} \tag{29}$$

$$r_{(i+1)} = -f'(x_{(i+1)}) \tag{30}$$

$$\beta^{\mathrm{FR}}_{(i+1)} = \frac{r^T_{(i+1)} r_{(i+1)}}{r^T_{(i)} r_{(i)}} \tag{31}$$

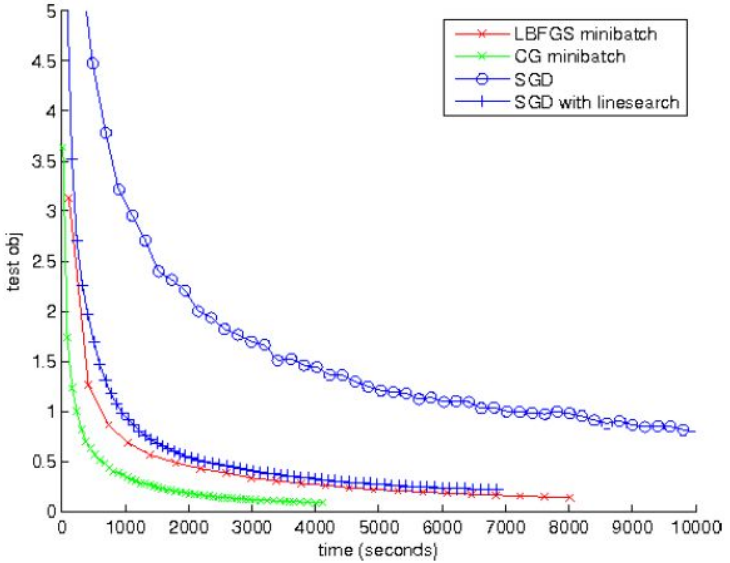
$$d_{(i+1)} = d_{(i)} + \beta_{(i+1)} d_{(i)} \tag{32}$$

Shewchuk (1994)

Nonlinear CGM and Autoencoders (Le et al. ICML 2011)

$$\min_{W,b,c} \sum_{i=1}^{m} \|\sigma(W^T \sigma(W x^{(i)} + b) + c) - x^{(i)}\|_2^2$$

- Nonlinear CGM converges faster than carefully tuned SGDs
- Nonlinear CGM perform better L-BFGS (quasi-Newton)
- CGM is more competitive than L-BFGS and SGDs in solving higher dimensional problems



Shewchuk (1994); Le et al. (2011)

ROUND UP ANALYSIS

Convergence

- Convergence for quadratics in finite steps
- May require mini-cycles in non-quadratics but works well with smaller n
- Reduced guarantee for convergence in non-linear case
- With a multi-modal non linear function, convergence to global minima is not guaranteed

Basic setting

- Useful for SPD
- Simplicity makes it easy to implement
- Very competitive deterministic method

Stochastic setting

- Many ML problems have a stochastic setting
- Stochastic CGM combines CGM and other stochastic Methods.

Round off errors

- Floating or round off errors may accumulate with number of iterations
- Pre-conditioning→faster convergence→ reduce accumulated error→better performance

WHAT NOW?

- CGM on average produces good results despite some limitations
- Hybrids schemes and extensions of CGM have comparable performance in practice.

Deb(2014); Hestenes and Stiefel (1952); Shewchuk (1994); Schraudolph and Graepel (2002)

CONCLUSION

CGM can outperform other algorithms in terms of speed of convergence, simplicity and cost efficiency.

However, these can be achieved according to several aspects such as linearity of the problem setting or the size of the matrix.

REFERENCES

- Deb, A. (2014). Lectures in numerical methods in civil engineering.
- Hestenes, M. R. and Stiefel, E. (1952). Methods of conjugate gradients for solving. Journal of research of the National Bureau of Standards, 49(6):409.
- Kimberly R., K. (2012). Implementing conjugate gradients with incomplete cholesky pre-conditioning in playa.
- Quoc V., L., Jiquan, N., Adam, C., Abhik, L., Bobby, P., and Andrew Y., N. (2011). On optimization methods for deep learning. International Conference on Machine Learning, 48(19).
- Schraudolph, N. and Graepel, T. (2002). Combining conjugate direction methods with stochastic approximation of gradients.
- Shewchuk, J. R. (1994). An introduction to the conjugate gradient method without the agonizing pain. Technical report.
- William, H. and Hongchao, Z. (2006). A survey of nonlinear conjugate gradient method. Pacific Journal of Optimization, 2(1).
- https://towardsdatascience.com/complete-step-by-step-conjugate-gradient-algorithm-from-scratch-202c07fb52a8