IE510 Applied Nonlinear Programming

Lecture 6: Effects of Eigenvalues

Ruoyu Sun

Feb 22, 2018. Week 6 Thursday.

Questions for Last Week

- **Q1**: Logistic regression is different from linear regression, although both are convex. In terms of optimization terminology, what are the differences? 1) Strongly convex v.s. (non-strongly) convex

2) Minimum attained for linear regression, not necessarily for logistic regression.

Additional: Changing Hessian v.s. fixed Hessian

- **Q2**: What are the two convergence behavior of logistic regression? The cause?

Linear, Sublinear

Cause: Non-separable, separable data.

Last Week: Case Study of Logistic Regression

- For 1-dim quadratic problems, three phases of convergence, for different stepsizes
- For 1-dim logistic regression, two different convergence behavior: linear and sublinear
  - Convex problem: worst-case $O(L/r)$ error after r iterations of GD
- How does condition number affect the speed?
  - ill-conditioned Hessian: at some iterate, $\lambda_{\max}/\lambda_{\min}$ is huge
  - changing Hessian: along the route, $\lambda'_{\min} s$ becomes too small
- Theoretical results are for the "worst-case" and "ideal case"
  - Last week data setting 1: $\min_w \lambda_{\min}(w) = 0$, but what matters is $\min_{w \in C} \lambda_{\min}(w)$ for some set C. Or "locally strongly convex"
  - Homework 2 problem 3: global L doesn't exist, some local L (in a level set C) can work

Today

- Effects of eigenvalues; and case study of linear regression
  - Note again: examples may be more complicated than theory
- After today's course, you will be able to
  - explain the practical behavior of GD for high-dim linear regression
  - explain the practical behavior of momentum (Tuesday 02/27)
- More advanced goal: analyze algorithms via eigenvalues

Outline

Review of Theories You Learned

- We study convex quadratic problems (rigorously, study linear regression)
  - every stationary point is globally optimal (so converge to global optima)
  - with lower bound (so $f(x^r)$ doesn't diverge) $\|Aw-b\|^2 \geq 0$;
  - achievable (so x^r doesn't diverge)

- When does GD "converge"?
  - Stepsize choice: $< 2/L$ with $L = \lambda_{\max}$

- Convergence rate?
  - $O(L/\epsilon)$ for convex problems $O(\frac{L}{\epsilon})$ iterations to achieve error ϵ.
  - $\kappa \log(1/\epsilon)$ iterations for strongly convex problems

The convergence rate result $O(\kappa)$ is a much simplified story...

Example: Salary Prediction

Kaggle dataset "Predict NHL Player Salary":
https://www.kaggle.com/datasets

- about 150 features $d = 150$
- about 600 samples $n = 600$

	A	B	C	D	E	F	G	H	I	J	K
1	Salary	Born	City	Pr/St	Cntry	Nat	Ht	Wt	DftYr	DftRd	Ovrl
2	925000	97-01-30	Sainte-Marie	QC	CAN	CAN	74	190	2015	1	18
3	2250000	93-12-21	Ottawa	ON	CAN	CAN	74	207	2012	1	15
4	8000000	88-04-16	St. Paul	MN	USA	USA	72	218	2006	1	7
5	3500000	92-01-07	Ottawa	ON	CAN	CAN	77	220	2010	1	3
6	1750000	94-03-29	Toronto	ON	CAN	CAN	76	217	2012	1	16
7	1500000	79-05-23	Strathroy	ON	CAN	CAN	70	192	1997	6	156
8	950000	90-11-21	Stockholm		SWE	SWE	71	185	2009	2	53
9	842500	93-07-28	Toronto	ON	CAN	CAN	70	183			
10	1250000	92-06-14	Scarborough	ON	CAN	CAN	72	214	2010	2	42
11	925000	93-04-27	Petawawa	ON	CAN	CAN	68	178	2011	7	201
12	800000	86-07-27	Lively	ON	CAN	CAN	73	195			
13	600000	92-04-09	Ile Bizard	QC	CAN	CAN	74	204	2010	3	89
14	1000000	88-03-17	Brandon	MB	CAN	CAN	72	200	2006	3	66
15	925000	96-06-20	Holland Land	ON	CAN	CAN	73	186	2014	1	4
16	950000	89-04-20	Red Deer	AB	CAN	CAN	72	195	2007	4	112
17	680000	94-01-06	Edmonton	AB	CAN	CAN	74	187	2012	4	105
18	590000	92-04-24	Oakville	ON	CAN	CAN	71	183	2011	7	209
19	650000	91-05-13	St. Catharine	ON	CAN	CAN	74	203	2009	3	70
20	2250000	86-04-28	Ostrava		CZE	CZE	74	235	2004	6	180
21	600000	90-10-08	Faribault	MN	USA	USA	76	210	2009	4	114
22	5000000	94-03-09	Vancouver	BC	CAN	CAN	73	215	2012	1	5

Use Linear Regression

Data preprocessing:

- Just pick numerical values
- Fill in the missing values by the average
- Normalize each feature (each column of data matrix X)
- Divide income y by 10^6, so typical range is about 0.9 to 10

reason: if $y \sim 10^6$, then $f(w) = \|Xw - y\|^2 \approx 10^6$,

figure like

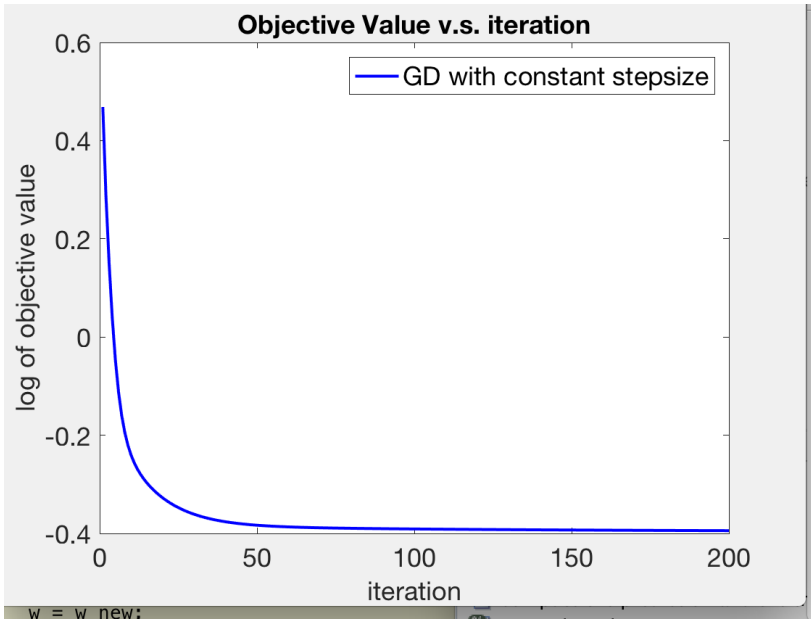
too large.

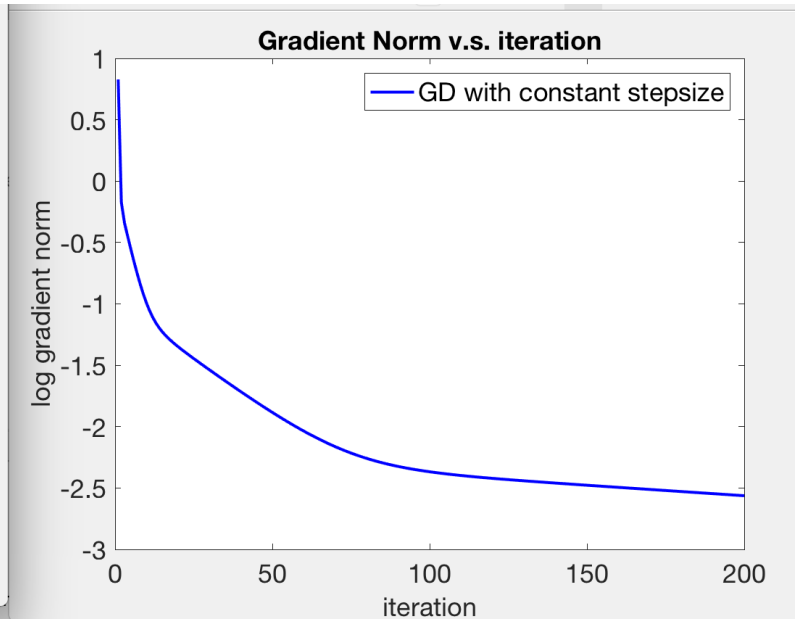
Problem and Algorithm:

- Solve $\min_w f(w) = 0.5\|Xw - y\|^2$;
- Report $f(w)/N$, so it should be $O(1)$
- Use GD with stepsize $1/L$, where $L = \lambda_{\max}(X^T X)$

Initial trial: Use a subset of the data $N = 10, d = 5$

GD Performance



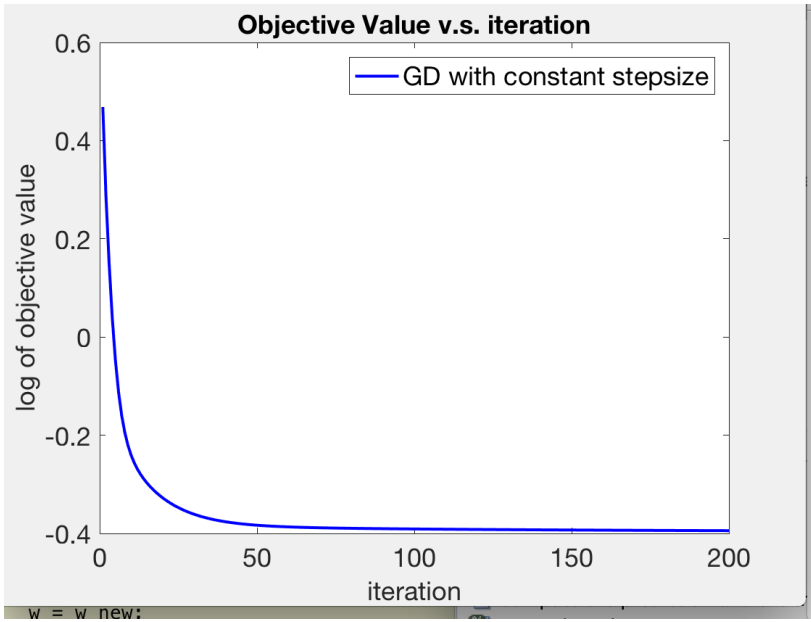


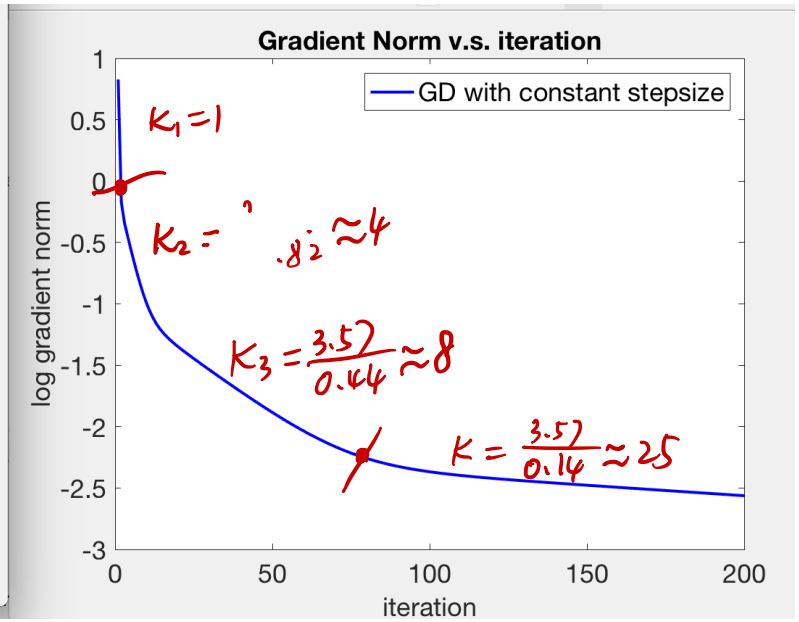
Observation: Seems like a sublinear rate of convergence?

Compute eigenvalues of $Q = X^T X$:
0.0139; 0.1497; 0.4417; 0.8206; 3.5742

- Strongly convex
- However, looks like sublinear rate of convergence

GD Performance





Observation: Seems like a sublinear rate of convergence?

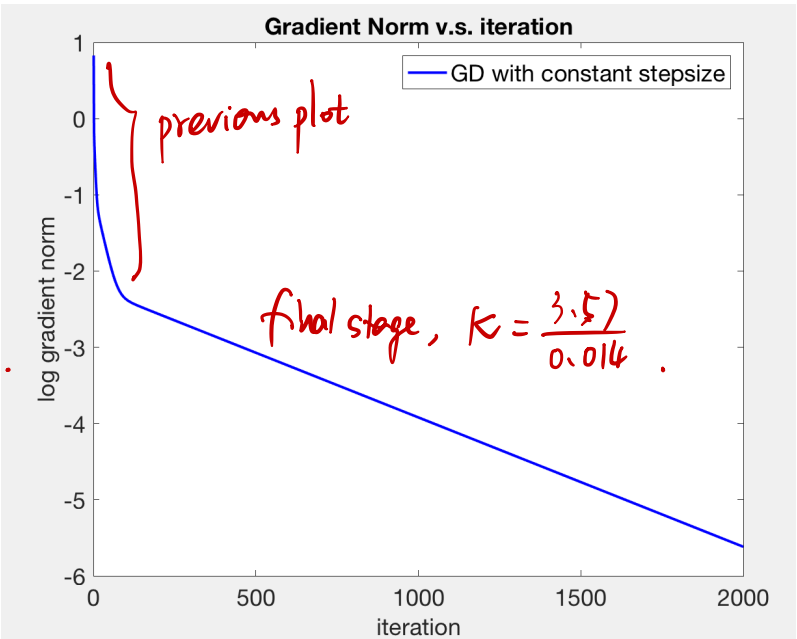
Compute eigenvalues of $Q = X^T X$:
0.0139; 0.1497; 0.4417; 0.8206; 3.5742

- Strongly convex
- However, looks like sublinear rate of convergence

Finally Linear Convergence

As in homework 2 problem 2, this is because GD goes through multiple stages.

It will finally converge in linear rate:



Question. Why the f-plot doesn't show "multiple-stages"?

Answer: $f(x) >> \lambda_{min}$, $\forall x$,
in fact, $f(x) \geq 0.3$ in this example.
so the multiple-stage behavior is hard to observe.
However, if you draw plot of
$f(x^k) - f^*$
you'll see similar "multi-stage" behavior.

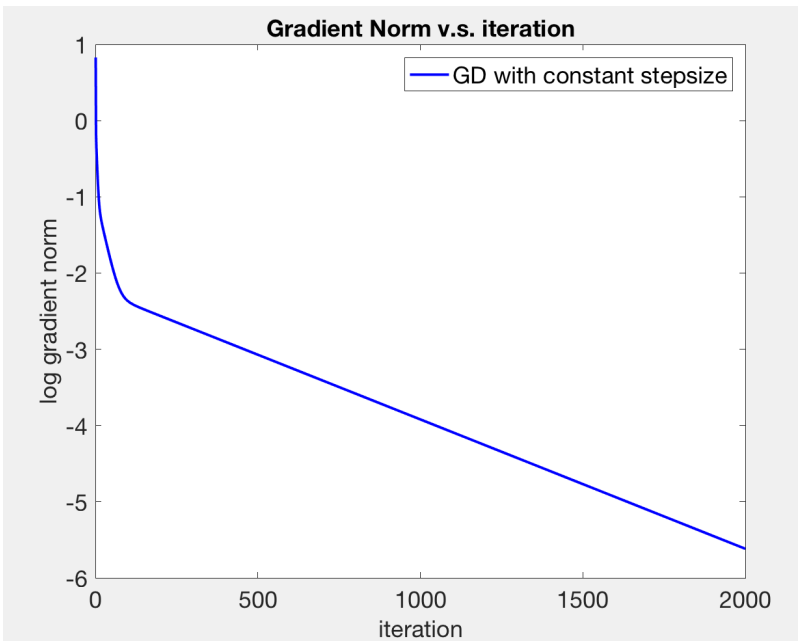
Predict # of iterations by condition number

$r = k \log 1/\epsilon$.

Condition number: $\kappa = 3.57/0.014 \approx 256$.

Sublinear: ×10 → ×100 → ×1000

error $= k \log 1/\epsilon$	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
predicted Iterations $f(x^r) - f^*$	590	1180	1770	2160	2750
		$=590\times2$	$=590\times3$	$=590\times4$	
true Iterations $\|\nabla f(x^r)\|$	10	50	500	1090	1680
			590	590	



Is the prediction accurate?

- Asymptotically accurate.

But not accurate for small ϵ.

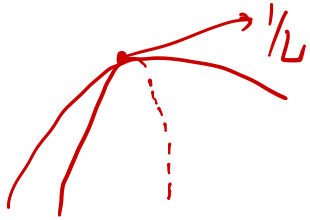
Practical Lesson

As a extension of homework 2 problem 2, we learned:

- GD will go through multiple stages in practice
- The theoretical complexity $\kappa \log(1/\epsilon)$ only captures the speed of the final stage
- Each stage has speed $\lambda_{\max}/\lambda_k$, and ends at roughly error λ_k
  - Need to consider "relative error"; usually we draw absolute error

Question. Why does "multi-stage" happen?

Answer: Stepsize $\alpha = 1/L = 1/\lambda_{max}$ is conservative.



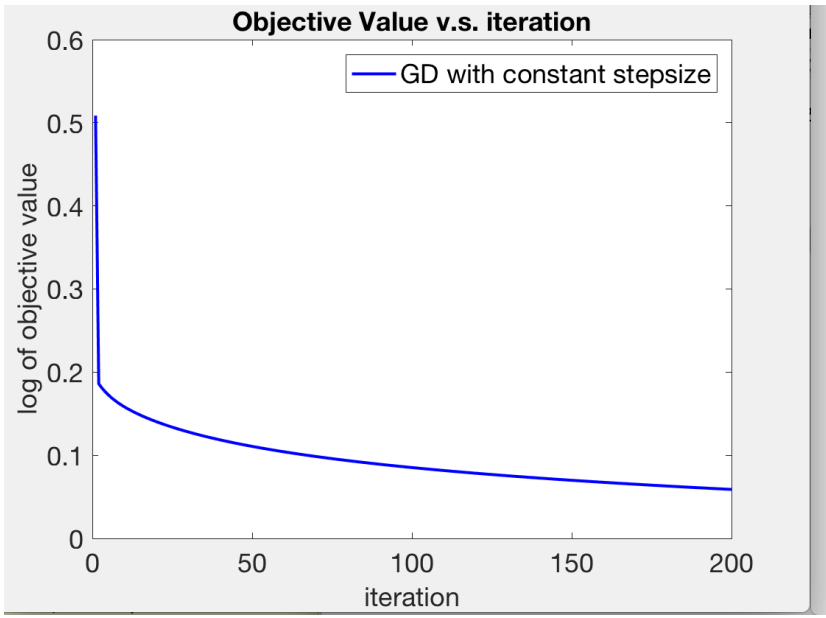
For different eigenmode, speed is different.

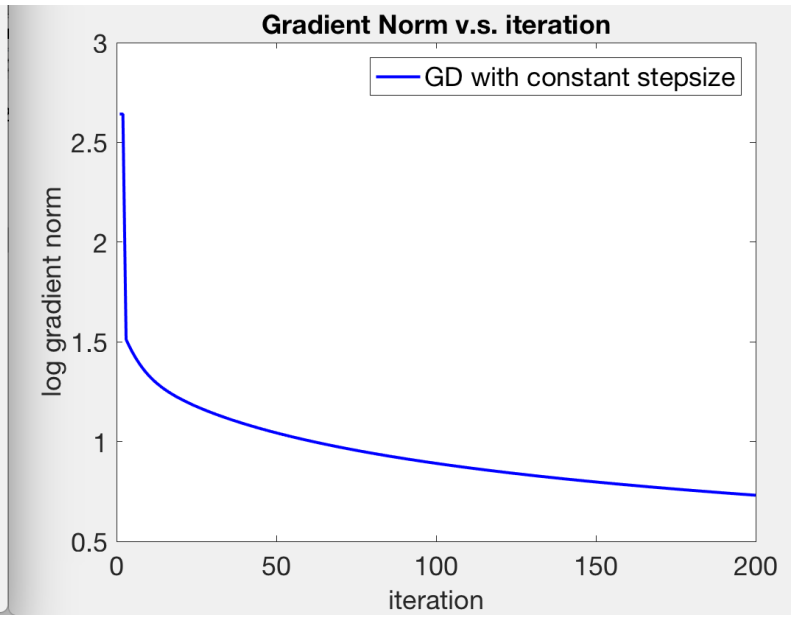
For eigenmode with λ_i, speed depends on $\frac{\lambda_{max}}{\lambda_i}$.

Original Data

Now we run GD on the whole dataset.

- $N = 612, d = 143$
- $L = 93$, $\alpha = 1/L = 0.0107$.





Question: Still multiple stages?

Most students say: yes.

Eigenvalue Distribution

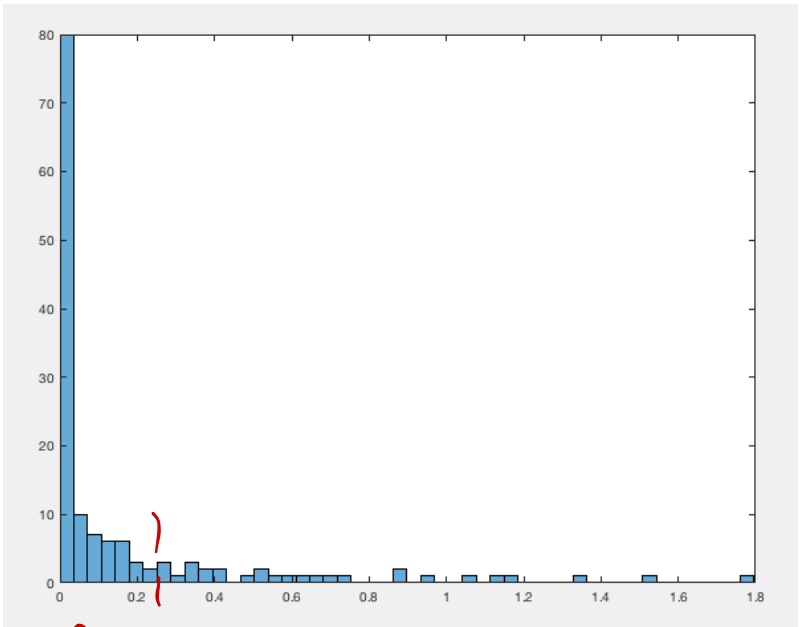
Eigenvalues

- Top 5: 93, 12.7, 6.1, 3.9, 3.3
- Bottom 8: $0, 0, 0, 0, 0, 0, 8 \times 10^{-7}, 5 \times 10^{-7}, 3 \times 10^{-7}$.

$\kappa = \frac{93}{3 \times 10^{-7}} \approx 3 \times 10^{8}$.

Non-strongly convex.

Comment: For quadratic problem, convergence rate depends on $\kappa = \frac{\lambda_{max}}{\lambda_{min}^{+}}$, where λ_{min}^{+} is the minimal non-zero eigenvalue.



Concentrate on 0~0.2

Figure: Histogram of all eigenvalues, except the top 5 $93, 12.7, 6.1, 3.9, 3.3$

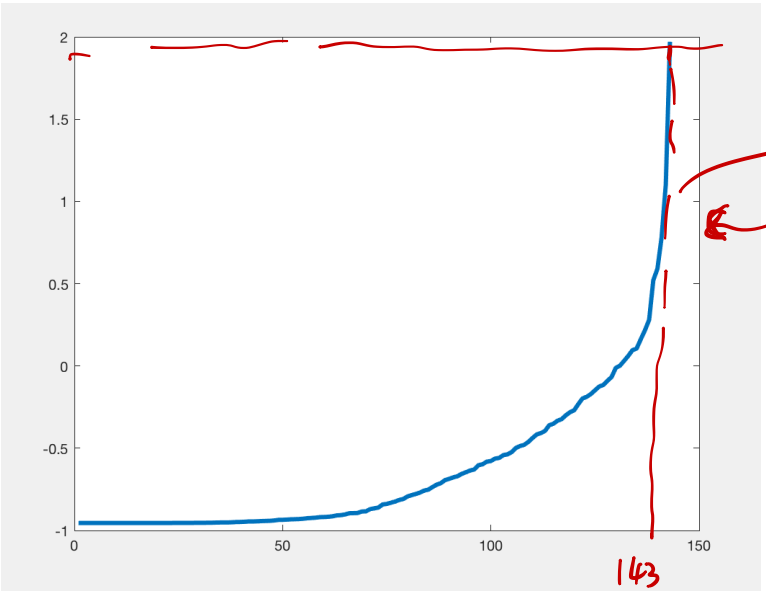
outliers of spectrum

Algorithm v.s. Eigenvalues

Observation: Algorithm plot highly correlated with eigenvalue distribution

Can produce algorithm plot purely based on eigenvalue plot!!

$10^{19} \approx 93$



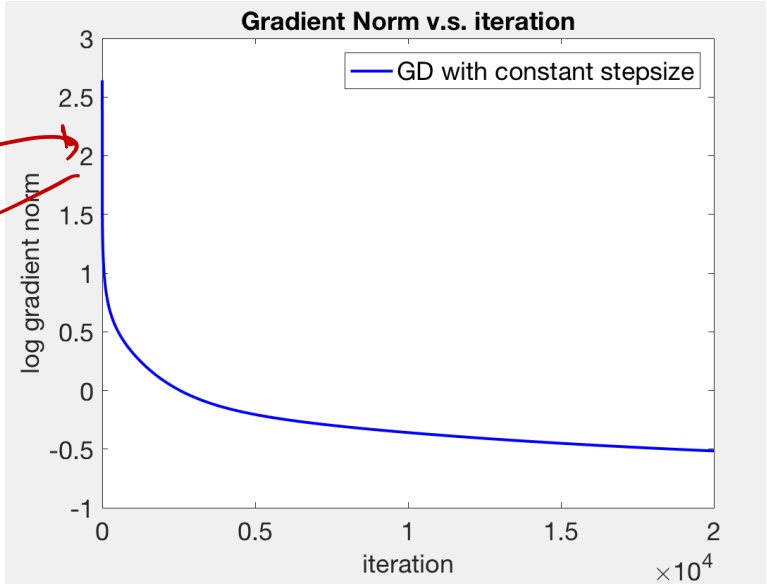


Figure: Left: eigenvalues; Right: gradient norm

How to get the figure?

Eigenvalues: $(\lambda_1, \lambda_2, \cdots, \lambda_{143})$, $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_{143}$.

Replace λ_i smaller than 0.1 by 0.1, get $(\hat{\lambda}_1, \hat{\lambda}_2, \cdots, \hat{\lambda}_{143})$

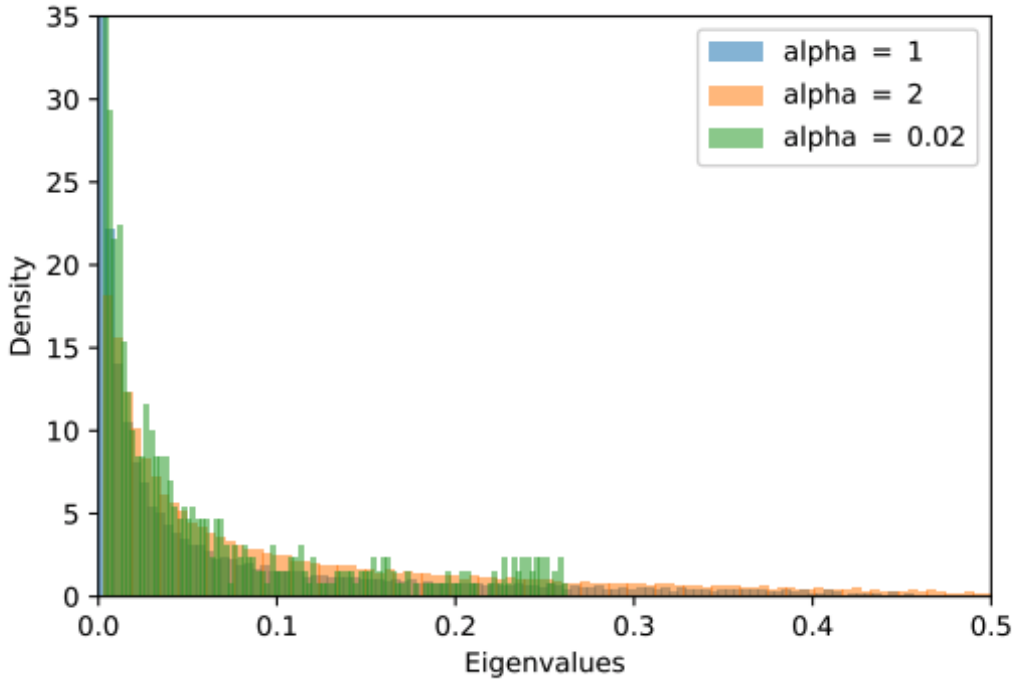
Plot $x = (1, 2, \cdots, 143)$ v.s. $y = (\hat{\lambda}_1, \hat{\lambda}_2, \cdots, \hat{\lambda}_{143})$.

Side: Eigenvalue Distribution

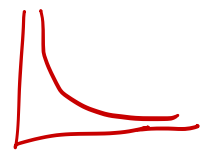
Eigenvalue distribution: central topic of random matrix theory.

The above distribution can be partially explained by existing theory;

see, e.g., Sagun et al.'17. Empirical Analysis of the Hessian of Over-Parametrized Neural Networks



Lesson:



This shape of eigenvalues is very common.
So this shape of algorithm plot is also common.

Figure: Sagun et al.'17 Fig 10b. The spectrum has outlier eigenvalues at 454.4, 819.5, and 92.7, respectively

Linear and Sublinear Rate

Insight on linear and sublinear:

- Because of the multi-stage behavior, linear rate looks like sublinear rate in early stages.
- For strongly convex problem, **finally** the linear rate will dominate
  maybe 100 years
- Too many eigenvalues + tend to zero gradually:
  - the linear rate will never dominate (in reasonable amount of time)
  - it will be always like sublinear

This is also true for 1-dim logistic regression

- in finite region, it should be linear rate,
- but it may exhibit sublinear rate throughout

Linear and Sublinear Rate

Insight on linear and sublinear:

- Because of the multi-stage behavior, linear rate looks like sublinear rate in early stages.
- For strongly convex problem, **finally** the linear rate will dominate
- Too many eigenvalues + tend to zero gradually:
  - the linear rate will never dominate (in reasonable amount of time)
  - it will be always like sublinear

This is also true for 1-dim logistic regression

- in finite region, it should be linear rate,
- but it may exhibit sublinear rate throughout

Practical Implication

What about a huge problem with 10^6+ samples, 10^4+ features?

- Feature engineering: create more features, so d can be huge too

Condition number? Usually huge, say, $> 10^6$.

If you want low-accuracy solution (e.g. in machine learning), iteration complexity $\kappa \log(1/\epsilon)$ is not an issue.

- Convergence behavior of GD roughly depends on eigenvalues above or around the accuracy

However, if you want high-accuracy solution, GD-type methods are bad... (e.g. in PDE)

2018.02.27. Week 7 Tues.

Quiz.

1) If eigenvalues are

(50, 0.2, 0.15, 0.1, 0.002, 0.0015, 0.001).

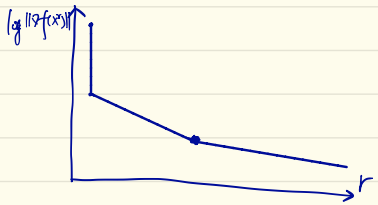
cluster 1 | cluster 2 | cluster 3

what will the figure be? (gradient norm or $f(x^k) - f(x^*)$)

2) Is

possible figure for GD with stepsize $1/L$, when solving convex quadratic problems?

Q1.



Remark: The iterate error $\|x^r - x^*\|$ will NOT behave like this.
See homework 2 problem 2.

Q2. NO. Consider $\min_x x^T Q x$, $Q = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix}$, $\lambda_1 \geq \lambda_2 \geq \lambda_3$, $x^0 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$.

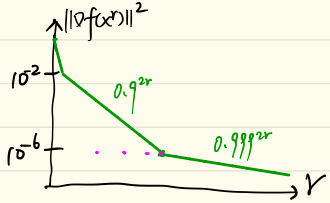
$$\|\nabla f(x^r)\|^2 = \|Q x^r\|^2 = \underbrace{(1-\alpha\lambda_1)^{2r}}_{=0}\lambda_1^2 + (1-\alpha\lambda_2)^{2r}\lambda_2^2 + (1-\alpha\lambda_3)^{2r}\lambda_3^2$$

$$= \left(1 - \frac{1}{L}\lambda_2\right)^{2r}\lambda_2^2 + \left(1 - \frac{1}{L}\lambda_3\right)^{2r}\lambda_3^2$$

e.g. $\lambda_1 = 1, \lambda_2 = 0.1, \lambda_3 = 10^{-3}$

$$= 0.9^{2r} \cdot 0.01 + 0.999^{2r} \cdot 10^{-6}$$

effective after 0.01 error, speed 0.9^{2r} — starting to be effective after error 10^{-6}, speed 0.999^{2r}



Observation: "low-error" stage has slower speed !!

∴ The mentioned figure is not possible under the assumptions

Outline

GD with Momentum

We solve this linear regression problem by **GD with momentum**

$$w^{r+1} = w^r - \alpha \nabla f(w^r) + \underbrace{\beta(w^r - w^{r-1})}_{\text{momentum}}.$$

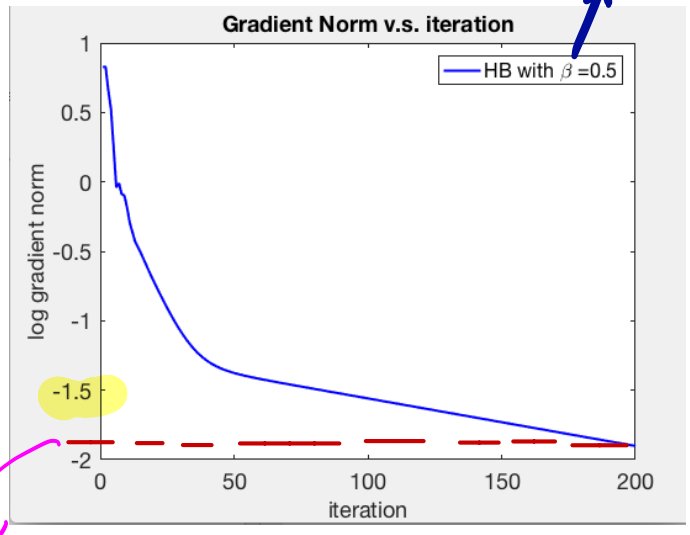
Algorithm parameters

- $\alpha = 1/L$, where $L = \lambda_{\max}(X'X)$
- $\beta \in (0, 1)$ and close to 1

Small data setting: $N = 10, d = 5$

Different β

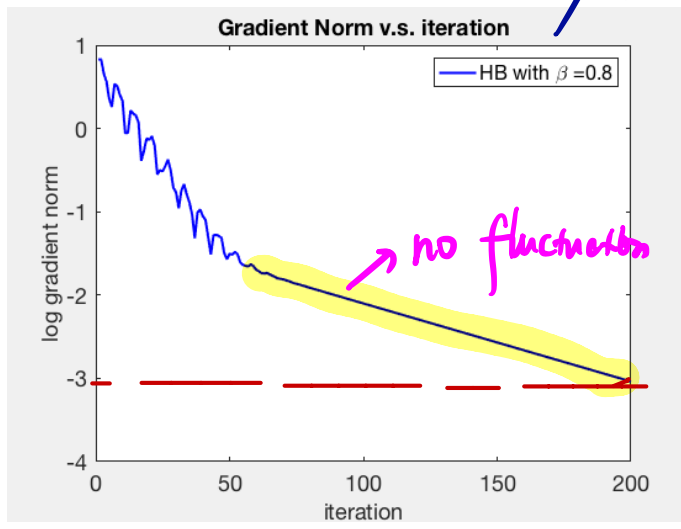
What do you observe?



$\beta=0.5$

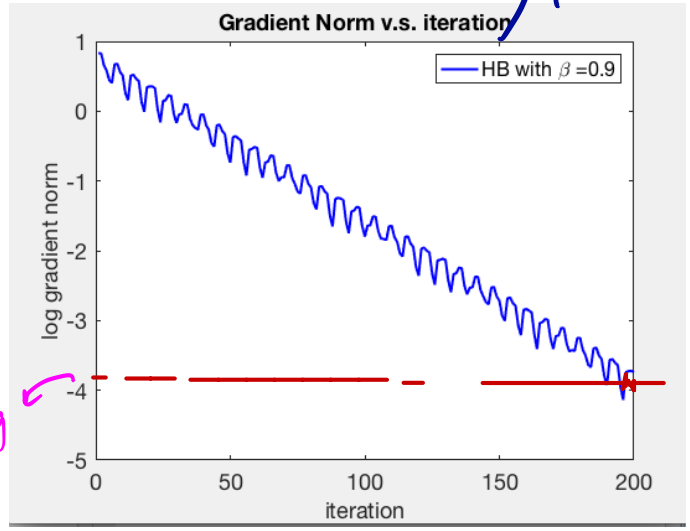
y-axis: $\log \|\nabla f(x^r)\|$

slow



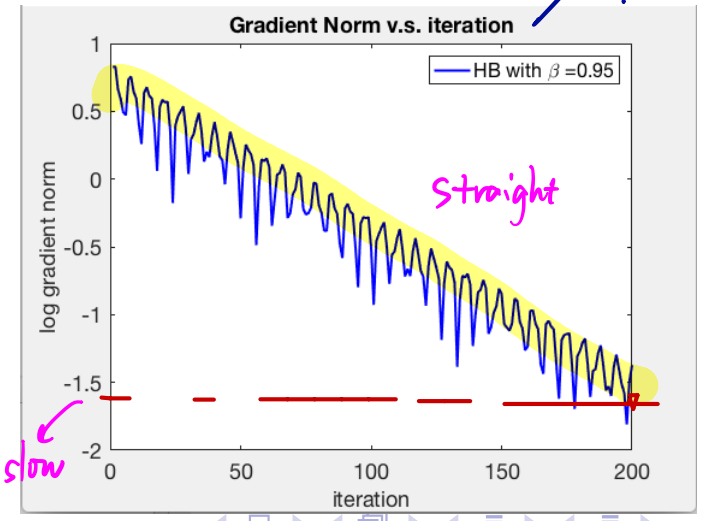
$\beta=0.8$

no fluctuation



$\beta=0.9$

fastest among these 4.



$\beta=0.95$

Straight

slow

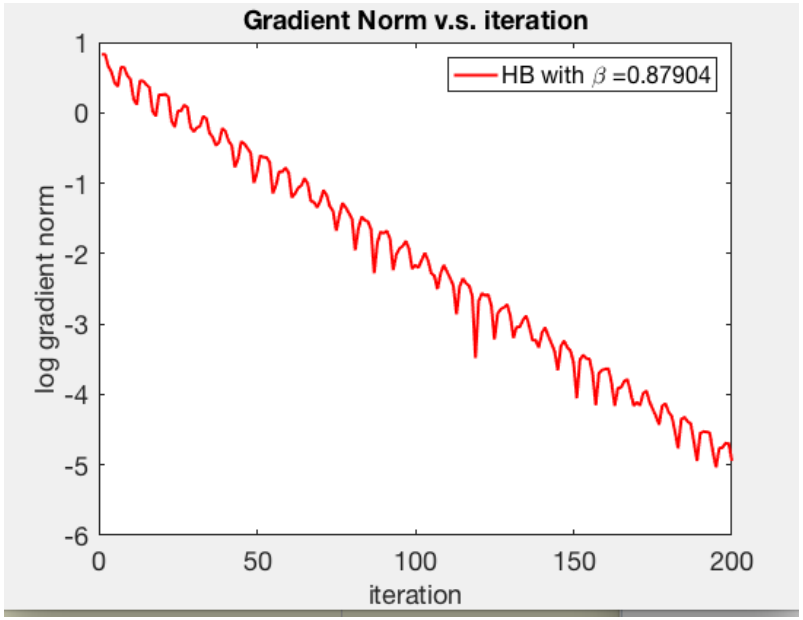
Optimal β

What is the optimal β?

According to theory, it is _____;

$(1-\frac{1}{\sqrt{\kappa}})^2$ $\quad$ ~~$1-\frac{1}{\kappa}$~~

$\alpha = 0.2798, \lambda_{\min} = 0.0139$



Sensitivity of parameter?

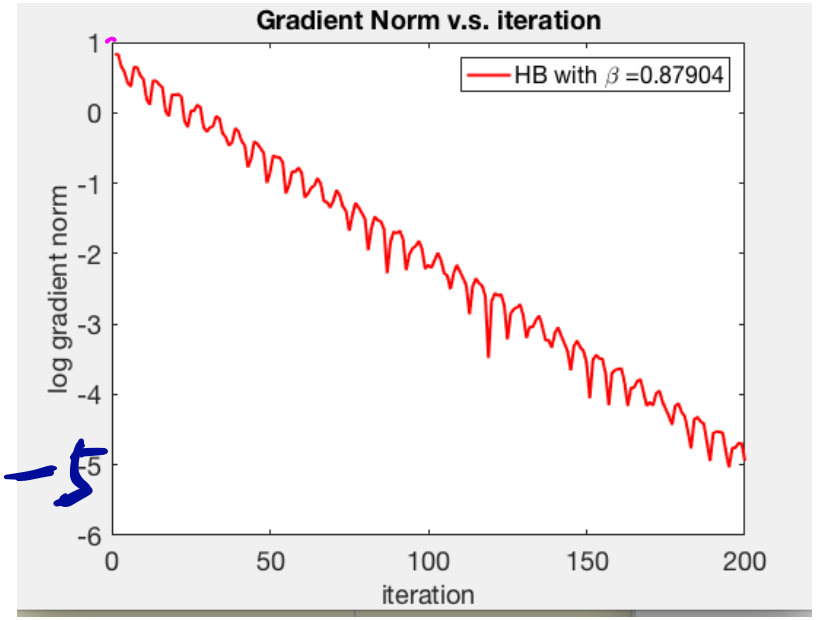
- The range _____ is good
- Conservative is ____ than aggresive β

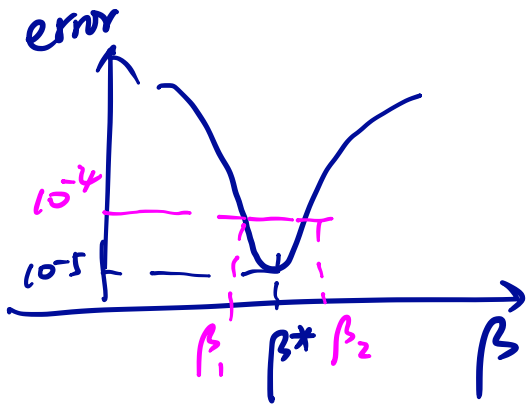
Optimal β

What is the optimal β?

According to theory, it is _____;

$\alpha = 0.2798, \lambda_{\min} = 0.0139$ $\beta = \left(1 - \frac{1}{\sqrt{k}}\right)^2 = 0.879.$





Comment:
Reinceforcemen learning
(e.g. AlphaGo)
sensitive to parameters

Optimal error 10^{-5} in 200 iterations.
- 10^{-4} is good enough.
- For what β, get error 10^{-4}?

Sensitivity of parameter?

- The range 0.85~0.90 is good
- Conservative is better than aggresive β

Remark 1. Why is sensitivity analysis of parameters useful?

- Guide hyper-parameter tuning accuracy.

Should you try $0.1, 0.2, 0.3, \cdots, 0.9, 0.95$, or try $0.1, 0.11, 0.12, 0.13, \cdots, 0.98, 0.99$?

Answer: At least for this problem, we know:

a) Try $0.1, 0.2, 0.3, \cdots, 0.9, 0.95$ is enough to achieve "good" error (lose 10^{-1} accuracy due to approximate search).

b) Even more efficiently, one should try

$\beta = 0.5,\ 0.8 = 1-\frac{1}{5},\ 1-\frac{1}{6},\ 1-\frac{1}{7},\ 1-\frac{1}{8},\ 1-\frac{1}{9},\ 1-\frac{1}{10} = 0.9,\ 1-\frac{1}{11},\ \cdots,\ 1-\frac{1}{20} = 0.95.$

Practical Tip: Try $\beta = 1-\frac{1}{q}$, where q is integer; typical $q=5$, $q=10$, $q=20$, $q=100$, etc.
$\beta = 0.8$, $\beta = 0.9$, $\beta = 0.95$, $\beta = 0.99$.

do bisection on q.

Remark 2 Why most blogs say "typical $\beta = 0.8, 0.9, 0.95, 0.99, \cdots$" ?

Answer: (a) $\beta^* \approx 1-\frac{2}{\sqrt{K}}$, so if $K=100$, $\beta^*=0.8$; if $K=400$, $\beta^*=0.9$, $\cdots$.

(b) To achieve error $10^{-3} \sim 10^{-5}$ in practice, due to the "multi-stage" analysis, you can view the "effective condition number" to be $10^3 \sim 10^5$.

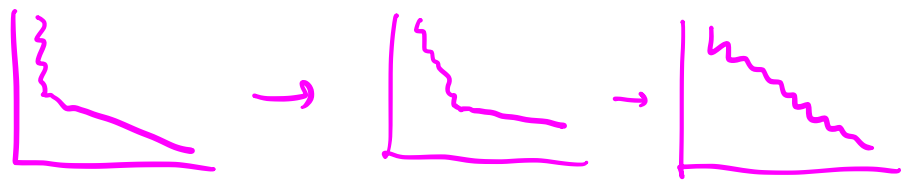
Comment: We can use parameter $\beta = 1-\frac{1}{q}$ because $1-\frac{1}{\beta} \approx \sqrt{K}$ affects the speed; for other algorithms, you may have to try linear scale of $\beta = 0.1, 0.2, 0.3, \cdots, 0.9$ first.

Practical Guidance

- What we have observed:
  - Optimal β (and larger): straight line, more fluctuation
  - Small β: multi-stage behavior, similar to GD

Empirical guidance: (when solving quadratic problems) in HB, assume $\alpha = 1/L$, then tune β till (for log of gradient norm):

- whole plot has fluctuation
- plot becomes straight



- The theoretical analysis of HB (partially in Lecture 3) can explain this Why straight line? Convergence rate for $\{x_i^r\}$ is β, whenever $1-\sqrt{\alpha\lambda_i} \le \sqrt{\beta} < 1$. Same rate β for all eigenmode i. Compare GD: rate $1-\alpha\lambda_i$ for eigenmode i.
- If you don't know theory, how to understand different behavior of HB method? Eigenvalues.

Outline

Preliminary knowledge:

Lecture 1 & hw 1: $x^{k+1} = Mx^k$ (*)

Convergence behavior depends on eig(M):

(i) if $\rho(M) \triangleq \max_i |\lambda_i(M)| < 1$, converge; if $\rho(M) > 1$, diverge; if $\rho(M) = 1$, undetermined.

(ii) If $\rho(M) < 1$, then closer to 1 means slower.

Matrix Recursion

The objective $f(w) = .5w^T Qw - w^T b + c$.

Write $w^{r+1} = w^r - \alpha \nabla f(w^r) + \beta(w^r - w^{r-1})$ as

$$y^{r+1} = My^r, \text{ where } M = \begin{bmatrix} I - \alpha Q + \beta I & -\beta I \\ I & 0 \end{bmatrix}.$$

Assume $b = c = 0$.

$$w^{r+1} = w^r - \alpha Q w^r + \beta(w^r - w^{r-1}) = (I - \alpha Q + \beta I) w^r - \beta \cdot w^{r-1},$$

Define $y^{r+1} \triangleq \begin{bmatrix} w^{r+1} \\ w^r \end{bmatrix} \in \mathbb{R}^{2n \times 1}$,

$$y^{r+1} = \begin{bmatrix} w^{r+1} \\ w^r \end{bmatrix} = \begin{bmatrix} I - \alpha Q + \beta I & -\beta I \\ I & 0 \end{bmatrix} \begin{bmatrix} w^r \\ w^{r-1} \end{bmatrix} = M \cdot y^r .$$

The behavior of HB is determined by eigenvalues of matrix M (for quadratic case)

Remark: For general quadratic problem ($b \neq 0$, $c \neq 0$),

$y^r \triangleq \begin{bmatrix} w^r \\ w^{r-1} \end{bmatrix} - \begin{bmatrix} w^* \\ w^* \end{bmatrix}$, where w^* is global-min.

Basic Plot for Complex Eigenvalue

M is non-symmetric, so may have complex eigenvalues.

Complex eigenvalue $\lambda = q e^{j\theta}$ leads to behavior like $q^r \cos(r\theta + \varphi)$,

q^r: decreasing; $\cos(r\theta+\varphi)$: fluctuation

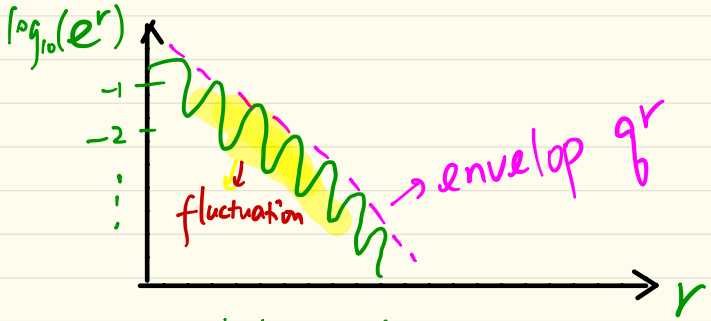


Figure: behavior of $e^r = q^r \cos(r\theta+\varphi)$

An analysis is not easy. See next pages (not required to know).

Appendix: How to apply eigenvalues of M to explain HB algorithm?

First of all, a precise analysis will take ≥ 5 pages, if ever possible.
We'll discuss the technical difficulty later; for now, consider a special choice

Consider an eigenvector v of M s.t. $Mv = \lambda v$, where $\lambda \in \mathbb{C}$ is complex.
We'd like to pick $y^0 = v$ to see the behavior of y^r, but v is a complex vector, so we can only pick $y^0 = Re(v) = \frac{1}{2}(v + \bar{v})$.

Now let's check the behavior of $\{y^r\}$.

$$y^1 = My^0 = M \cdot \frac{1}{2}(v + \bar{v}) = \frac{1}{2}(Mv + M\bar{v})$$
$$= \frac{1}{2}(\lambda v + \bar{\lambda}\bar{v});$$

Continue the process, we get $y^r = \frac{1}{2}(\lambda^r v + \bar{\lambda}^r \bar{v}) = Re(\lambda^r v)$

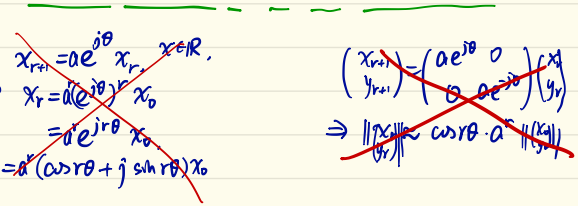
Suppose $\lambda = q e^{j\theta}$, where $q > 0$, $\theta \in \mathbb{R}$. Then $y^r = Re(q^r e^{jr\theta} \cdot v)$

Let's check the first element: assume $v_1 = p e^{j\varphi}$, $p > 0$, $\varphi \in \mathbb{R}$, then

$$y_1^r = Re(p \cdot q^r e^{j(r\theta + \varphi)}) = p \cdot q^r \cos(r\theta + \varphi).$$

q^r: decreasing in rate $q = |\lambda|$; $r\theta + \varphi$: fluctuation due to θ = phase of λ.

Wrong analysis



General Case (difficulties)

We only analyzed one case: y^0 is real part of some eigenvector of M (and one element y^r_1)

What about general y^0 ? It's quite difficult to analyze rigorously.

One major difficulty is that M may not be diagonalizable, so we may not be able to express y^r by the eigenvectors of M. To resolve the issue, we may need to use Jordan form, an advanced topic.

Another difficulty: $y^{r+1} = My^r$ decribes the behavior of $\begin{pmatrix} w^r \\ w^{r-1} \end{pmatrix}$ expressed in terms of the eigenmodes of M (i.e. eigenvectors of M), but our metric is $\|\nabla f(w^r)\| = \|Qw^r\|$ when $f(w) = \frac{1}{2} w'Qw$.

The behavior of $\|y^r\|$ may be different from $\|\nabla f(w^r)\|$.

Anyhow, the simulation shows that eigenvalues of M can mostly explain the behavior of $\|\nabla f(w^r)\|$

Eigenvalues of Update Matrix M

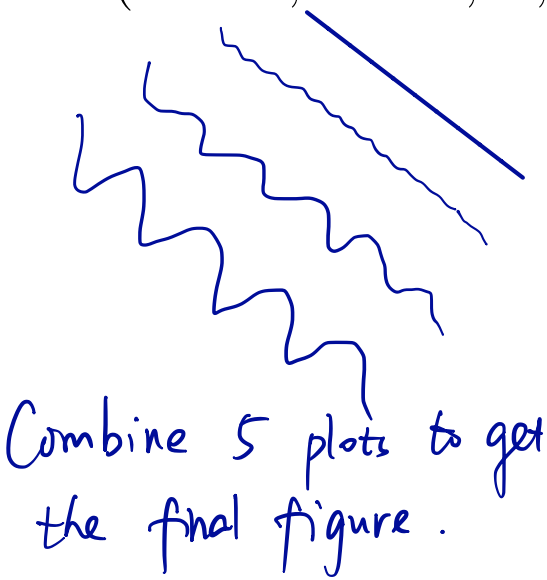
$\beta = 0.879$ (optimal β).

Eigenvalues of M are (2d)×(2d) = 10×10.

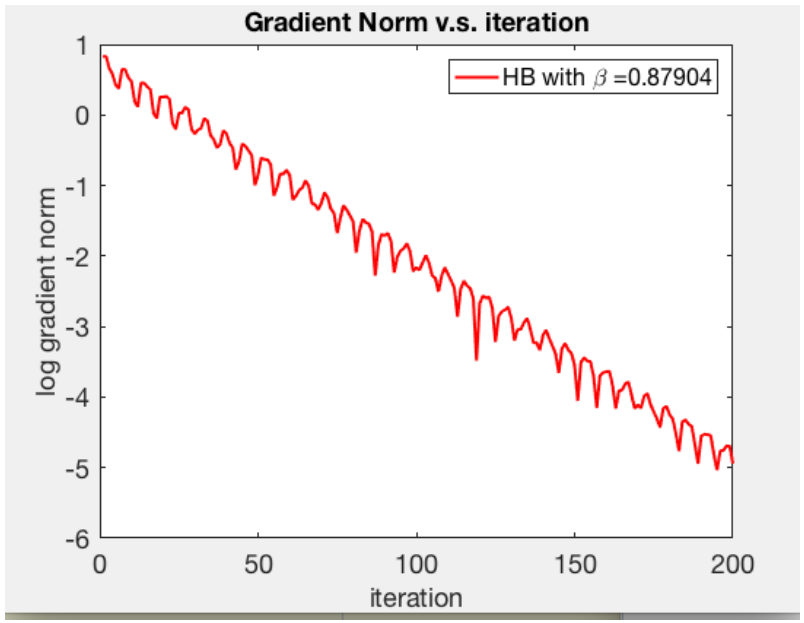
$(0.44 \pm 0.82i,\ 0.82 \pm 0.44i,\ 0.88 \pm 0.33i,\ , 0.92 \pm 0.19i, 0.9376, 0.9376)$

Absolute eigenvalues of M are

$(0.9376, 0.9376, ..., 0.9376, 0.9376, 0.9376,\ 0.9376)$



Combine 5 plots to get the final figure.



How to explain the figure using the eigenvalues?

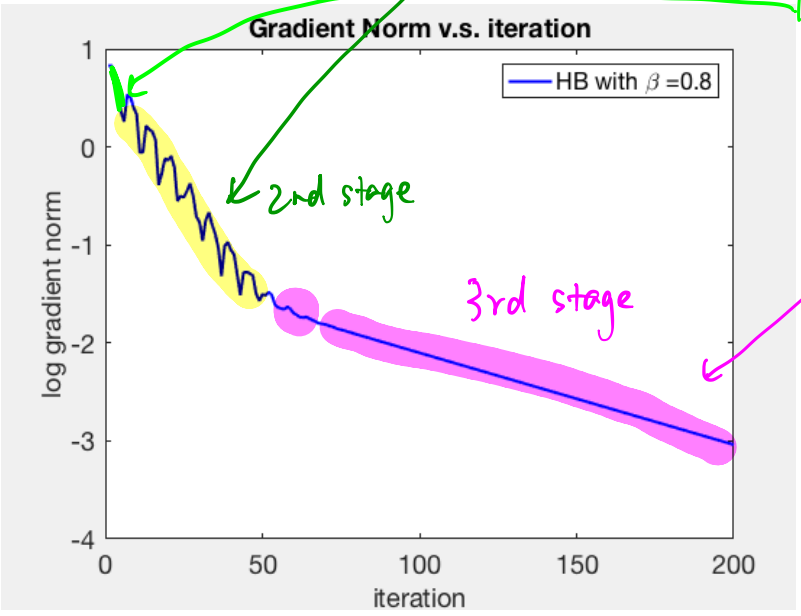
- 1 stages: since all eigenvalues have the same magnitude.
- Fluctuation: complex eigenvalues

Eigenvalues of Update Matrix M

$\beta = 0.8$.

Eigenvalues of M are
$(0.4 \pm 0.8i,\ 0.79 \pm 0.43i,\ 0.84 \pm 0.31i,\ , 0.88 \pm 0.17i, 0.82\ , 0.98)$

Absolute eigenvalues of M are $(0.89, 0.89, ..., 0.89, 0.89, 0.82,\ 0.98)$

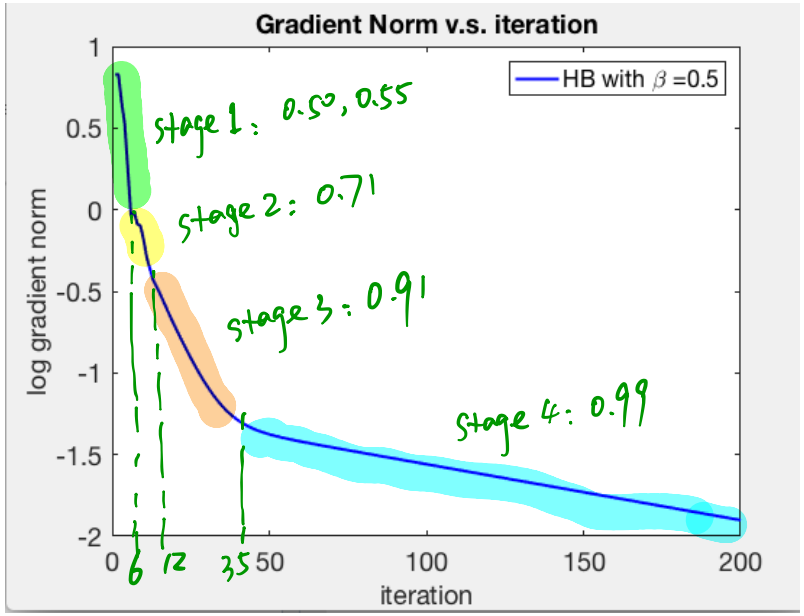


How to explain the figure using the eigenvalues?

- 3 stages: $\{0.82\}$ $\{0.89, 0.89, ..., 0.89\}$ $\{0.98\}$, 3 clusters
- Fluctuation: Only in 2nd stage, since 0.89 corresponds to complex eigenvalues

Eigenvalues of Update Matrix M

$\beta = 0.5$.



Quick exercise:

$a_r = 0.7^r$, how fast?

How long is the "0.7"-stage?

Answer: $\frac{1}{1-0.7}\log\frac{1}{\varepsilon} \approx 3\log\frac{1}{\varepsilon}$.

Length of each stage (roughly)

Stage	λ_i	$1/(1-\lambda_i)$	length
1	0.5	2	$2C_1$
2	0.71	3.3	$3.3C_2$
3	0.91	11	$11C_3$
4	0.99	100	$100C_4$

Eigenvalues of M are

$(0.25 \pm 0.66i,\ 0.64 \pm 0.31i,\ 069 \pm 0.16i, 0.50\ 0.55,\ ,0.91, 0.99)$

here C_1, C_2, C_3, C_4 are related to $\log\frac{1}{\varepsilon}$, so can be viewed as constant (precise values are not constant; see hw2 problem 2(c))

Absolute eigenvalues of M are

$$(0.71, ..., 0.71, 0.71, 0.50\ , 0.55,\ 0.91, 0.99)$$

$$(\pm 69^\circ,\quad \pm 26^\circ,\quad \pm 13^\circ,\quad 0,\quad 0,\quad 0,\quad 0)$$

Why little fluctuation? The second stage (the only one with fluctuation) is too short.

Tip: Length of a stage is proportional to $\frac{1}{(-\lambda_i}$.
Slope of a stage is proportional to $(1-\lambda_i)$

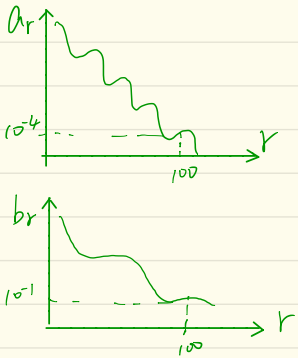
Tip on understanding different stages: Decomposition.

Suppose M has two pairs of eigenvalues $\lambda_{1,2} = q e^{\pm j\theta}$, $\lambda_{3,4} = p e^{\pm j\varphi}$.

Write two sequence

$$a_r = \mathrm{Re}\{q^r e^{\pm jr\theta}\}$$
$$b_r = \mathrm{Re}\{p^r e^{\pm jr\varphi}\},$$

Sum up the two figures, you get (approximately) the figure of $\|y^r\|$, where $y^{r+1} = My^r$.



Nesterov's Accelerated Method

We solve the linear regression problem by Nesterov's accelerated method.

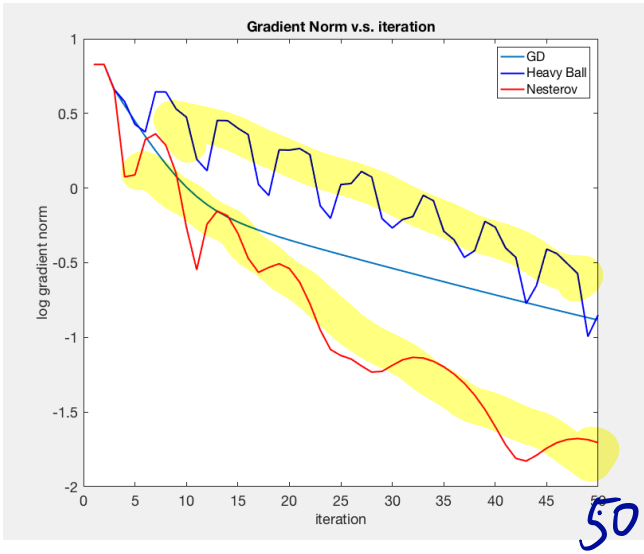
$$\begin{cases} y^r = x^r + \beta_r(x^r - x^{r-1}), \\ x^{r+1} = y^r - \alpha_r \nabla f(y^r). \end{cases} \tag{1}$$

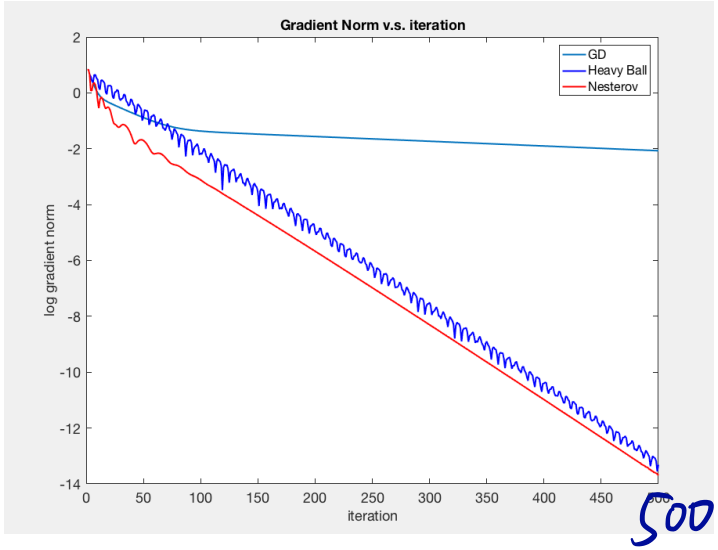
Stepsize $\alpha = 1/L$.

Try different β. (at home)

Comparison: Nesterov momentum and momentum

Using Optimal parameters for Nesterov ($\beta = 0.8825$), and HB ($\beta = 0.879$)





Observations:

- Nesterov's method is faster than GD in early stage, and similar to HB in later stages
- Nesterov's method has less fluctuation than HB

Outline

Summary: Specific Algorithms

The following observations are for quadratic problems; some of them may be generalized.

GD: has piecewise linear plot (multiple stages).

HB: the optimal plot is straight with fluctuation.

Nesterov: mixture of GD and HB: multiple stages, and fluctuation in early stage.

All the above observations can be supported by theory!

Summary: Effects of Eigenvalues

For quadratic functions, the performance of algorithms usually depends on the "update matrix". M. (not Q)

- Need to write the algorithm as matrix recursion.

Effects of eigenvalues on performance include:

- Worst-case rate: depends on condition number (with ∞ # of iterations)

slope $\sim 1-\lambda_i$, length of the stage $\sim \frac{1}{1-\lambda_i}$.

- Usually multiple stages, depends by magnitude of eigenvalues.

- Fluctuation depends on the phases of eigenvalues.

Tip: Use eigenvalues to study algorithms in your project (for small case first).

Further reading and thought

How to generalize to non-quadratic case? Nonconex case?

- HInt: If Hessians at every point is known, the behavior of algorithm can be largely explained.

But much more complicated than the class. Take 4 more classes at least to discuss, if not more ...

Related papers:

- Hessian of neural-nets (figure similar to page 14 of this lecture).

  Sagun et al. "Empirical Analysis of the Hessian of Over-Parametrized Neural Networks". (Facebook)

- Exploring fluctuation to improve Nesterov's method:

  Donoghue,Candes, "Adaptive Restart for Accelerated Gradient Schemes (Stanford)