METODE GAUSS–NEWTON

Turunan Matematis, Struktur Matriks, dan Analisis Konvergensi



Mohammad Jamhuri · Sri Harini · Andy Irawan Juhari · Mutiara Alvi · Natasyah Asviana

Penerbit Ediide Infografika

2026

Metode Gauss–Newton

Turunan Matematis, Struktur Matriks, dan Analisis Konvergensi

Copyright © Mohammad Jamhuri, dkk., 2026
xviii + 165 hlm, 14,8 × 21 cm

Penulis	: Mohammad Jamhuri, Sri Harini, Andy Irawan, Juhari, Mutiara Alvi, Natasyah Asviana
Editor	: Muhammad Khudzaifah, M.Si
Cetakan Pertama	: Oktober 2026
ISBN	: 978-634-7866-01-1

Diterbitkan pertama kali oleh:
Penerbit Ediide Infografika
Jl. Joyo Agung III, Blok J-21 Villa Tlogomas, Malang
Email: penerbit.ediide@gmail.com
Website: penerbitan.ediide.com

Anggota IKAPI Jawa Timur
No. 242/JTI/2020

All Rights Reserved © 2026

Hak Cipta Dilindungi Oleh Undang-undang.
Buku ini dilisensikan di bawah **Creative Commons Attribution 4.0 International (CC BY 4.0)**.

Anda bebas menyalin, mendistribusikan ulang, dan mengadaptasi materi ini dalam media atau format apa pun untuk tujuan apa pun, selama mencantumkan atribusi yang sesuai kepada para penulis.

Informasi lisensi: https://creativecommons.org/licenses/by/4.0/



Kode MATLAB dan Python dalam buku ini dilisensikan di bawah Lisensi MIT. Dataset publik yang dipakai dalam eksperimen tetap tunduk pada ketentuan penyedianya masing-masing.

Prakata

Buku ini ditulis untuk pembaca yang ingin memahami metode Gauss–Newton dari dasar, bukan sekadar menggunakannya sebagai perintah siap pakai. Selain itu, buku ini disusun sebagai pengantar yang runtut menuju pemahaman optimisasi nonlinear serta penerapannya dalam regresi, klasifikasi, dan jaringan saraf kecil. Sasaran utamanya ialah mahasiswa, dosen, dan praktisi yang telah mengenal kalkulus elementer, tetapi belum memiliki latar belakang aljabar linear atau optimisasi yang kuat.

Pembahasan dimulai dari satu gagasan sederhana. Sebuah model menghasilkan prediksi, sedangkan data menyediakan nilai pengamatan. Selisih keduanya membentuk residu. Dari sini, pembaca diajak melihat mengapa gradien, matriks Jacobian, persamaan normal, dan aproksimasi Hessian muncul secara alami. Setiap konsep diperkenalkan ketika diperlukan oleh langkah berikutnya.

Buku ini juga menempatkan eksperimen sebagai bagian dari penalaran. Contoh berukuran kecil dipakai untuk membedah perhitungan. Dataset publik digunakan untuk menunjukkan perilaku metode pada regresi, klasifikasi biner, klasifikasi multikelas, dan jaringan saraf berukuran kecil. Potongan kode MATLAB yang ringkas disertakan dalam kotak kode, tetapi kode lengkap untuk eksperimen data publik tidak dicetak di badan buku. Sebagai gantinya, prosedur eksperimen disusun bertahap agar pembaca dapat memperoleh hasil yang sebanding dengan perangkat lunak yang tersedia.

Gauss–Newton bukan metode yang berdiri sendiri. Metode ini berhubungan erat dengan dekomposisi QR dan SVD, Levenberg–Marquardt, IRLS, informasi Fisher, dan propagasi balik. Hubungan tersebut menjadi benang merah buku ini. Harapannya, setelah menyelesaikan pembahasan, pembaca tidak hanya mampu menghitung satu langkah algoritma, tetapi juga dapat menilai kapan langkah itu layak dipercaya.

Buku ini dirilis sebagai buku akses terbuka di bawah lisensi CC BY 4.0. Siapa pun boleh membaca, membagikan, dan mengadaptasinya untuk perkuliahan maupun keperluan lain, asalkan mencantumkan sumbernya. Kritik dan saran dari pembaca sangat kami harapkan untuk perbaikan edisi berikutnya.

Malang, September 2026

Penulis

Cara Menggunakan Buku

Urutan bab dirancang secara bertahap. Bab 1–3 membangun masalah dan menurunkan algoritma. Bab 4–8 membahas kestabilan, konvergensi, redaman, dan pembobotan. Bab 9–14 membawa metode ke regresi, klasifikasi, dan jaringan saraf. Bab 15–16 menjelaskan hubungan statistik serta jaminan konvergensi global. Dua bab terakhir merangkum keputusan praktis dan menyediakan latihan.

Kotak-kotak di dalam buku memiliki fungsi yang berbeda. Intuisi memberi gambaran sebelum rumus. Contoh Angka menguraikan perhitungan kecil. Catatan Aljabar Linear menjelaskan perangkat matriks yang sedang dipakai. Langkah Eksperimen menyajikan prosedur sebelum hasil ditampilkan. Pendalaman Teknis menandai materi yang dapat dilewati pada pembacaan pertama.

Peta baca

- **Jalur intuisi.** Baca Bab 1–3 tanpa bukti, lalu lanjutkan ke Bab 5, Bab 9, dan Bab 17. Gunakan glosarium ketika menemukan istilah yang belum akrab.
- **Jalur praktisi.** Tambahkan Bab 4 tentang penyelesaian stabil, Bab 7 tentang Levenberg–Marquardt, dan Bab 10–14 tentang klasifikasi serta jaringan saraf.
- **Jalur lengkap.** Baca seluruh bab dan lampiran. Jalur ini sesuai untuk perkuliahan, penelitian, atau pembaca yang ingin memeriksa asumsi setiap teorema.

Simbol ▲ menandai uraian teknis. Pada pembacaan pertama, kita dapat mengambil gagasan utamanya lalu melanjutkan ke subbab berikutnya.

Konvensi Istilah dan Notasi

Istilah Indonesia digunakan sebagai istilah utama. Istilah Inggris dicantumkan ketika pertama kali diperlukan, terutama jika istilah tersebut lazim muncul dalam pustaka atau perangkat lunak. Buku ini memakai kata *residu* untuk selisih $y_i - f_i$. Padanan Inggris untuk *lapisan tersembunyi*, *propagasi maju*, *propagasi balik*, dan *pencarian garis* ialah *hidden layer*, *forward pass*, *backpropagation*, dan *line search*.

Vektor ditulis tebal, misalnya $\boldsymbol{r}$ dan $\boldsymbol{\theta}$. Matriks memakai huruf kapital, misalnya J dan H. Indeks i menyatakan pengamatan, sedangkan indeks k menyatakan iterasi. Simbol yang lebih khusus dijelaskan kembali ketika pertama kali muncul. Bilangan desimal ditulis dengan titik, misalnya 0.538, agar sama dengan keluaran perangkat lunak numerik dan kotak kode MATLAB.

Catatan Reproduksibilitas

Setiap eksperimen mencantumkan sumber data, pembagian data, nilai awal, aturan pembaruan, dan kriteria berhenti sebelum gambar atau tabel hasil. Jika suatu pustaka menyediakan fitur yang telah diskalakan, buku mengikuti bentuk bawaan yang disebutkan pada langkah eksperimen. Hasil pengacakan data juga bergantung pada jenis generator bilangan acak, sehingga generator yang dipakai pada eksperimen acuan disebutkan pada langkah eksperimen. Angka hasil dapat berbeda sedikit akibat versi perangkat lunak, pembulatan, atau pustaka aljabar linear. Perbedaan kecil tersebut tidak mengubah pola konvergensi yang sedang diperiksa.

Daftar Isi

Prakata **i**

Cara Menggunakan Buku **iii**

Daftar Gambar **xiii**

Daftar Tabel **xvii**

1 Pendahuluan **1**

1.1 Rumusan Masalah 1

1.2 Peta Studi Kasus dan Replikasi 3

1.3 Pemanasan: Tiga Titik dan Garis Lurus 4

2 Menuju Kondisi Optimal: Gradien dan Matriks Jacobian **9**

3 Dari Newton Klasik ke Aproksimasi Gauss–Newton **15**

3.1 Metode Newton untuk Optimisasi 15

3.2 Gagasan Gauss–Newton: Membuang Suku Kedua 18

3.3 Menurunkan Persamaan Normal Gauss–Newton 21
3.4 Algoritma Iteratif 24

4 Menyelesaikan Persamaan Normal secara Stabil 27
4.1 Dekomposisi Nilai Singular (SVD) dan Bilangan Kondisi 27
4.2 Alternatif Stabil: Dekomposisi QR 29
4.3 Kasus Kekurangan Peringkat: Solusi melalui SVD 31

5 Menjamin Kemajuan Setiap Langkah: Arah Turun 33
5.1 Apa yang Dimaksud dengan Arah Turun? 33
5.2 Arah yang Benar Belum Tentu Berarti Langkah Penuh 34
5.3 Prosedur Pemeriksaan pada Setiap Iterasi 36

6 Seberapa Cepat Metode Ini Konvergen? 37
6.1 Dua Besaran Kunci: Ketidaklinearan dan Kekondisian 37
6.2 Membaca Laju Konvergensi dari Data Iterasi 38
6.3 Teorema Konvergensi Lokal 39
6.4 Apa yang Perlu Dilaporkan dalam Eksperimen 40

7 Mengatasi Ketidakstabilan: Regularisasi Levenberg–Marquardt 43
7.1 Strategi Praktis Menyesuaikan λ_k 47
7.2 Konvergensi Global Levenberg–Marquardt 51
7.3 Eksperimen: Marquardt versus Nielsen pada MLP 52
7.4 Alternatif: Dogleg dan Persamaan Sekuler Moré 55

8 Generalisasi ke Kuadrat Terkecil Berbobot 61

8.1 Mengapa Data Tidak Selalu Boleh Diperlakukan Sama? 62
8.2 Dari Residu Asli ke Residu Efektif 63
8.3 Langkah Perhitungan yang Dapat Diulang 64

9 Contoh Numerik Lengkap 67

9.1 Membaca Kurva Pembelajaran: Gauss–Newton versus Penurunan Gradien 70
9.2 Eksperimen Publik: SLP untuk Regresi Diabetes 72

10 Penerapan pada Klasifikasi Biner: Regresi Logistik dan IRLS 75

10.1 Mengapa Kuadrat Terkecil Tidak Lagi Tepat 75
10.2 Turunan Pertama dan Kedua dari Kerugian Entropi-Silang . . . 78
10.3 Menurunkan Metode Newton untuk Regresi Logistik 79
10.4 Persamaan Newton sebagai Kuadrat Terkecil Berbobot: Algoritma IRLS 81
10.5 Contoh Numerik Singkat 82

11 Eksperimen Empiris: Prediksi Keselamatan Penumpang Titanic 87

11.1 Penyiapan Data dan Model 87
11.2 Hasil: Kecepatan Konvergensi 89
11.3 Hasil: Akurasi dan Koefisien Model 90

12 Klasifikasi Multikelas dengan Softmax dan Perceptron Satu Lapisan 91

12.1 Dari Sigmoid ke Softmax 91
12.2 Kerugian Entropi-Silang Multikelas dan Turunannya 93

12.3 Sekali Lagi, Gauss–Newton Menjadi Newton Eksak 96
12.4 Persamaan Normal Multikelas dan Algoritma IRLS 96
12.5 Contoh Numerik: Tiga Kelas 98
12.6 Eksperimen Publik: Iris dan Softmax 100

13 Jaringan Berlapis: MLP dengan Satu Lapisan Tersembunyi 103

13.1 Model MLP sebagai Wujud Konkret dari $f(\boldsymbol{x}, \boldsymbol{\theta})$ pada Bab 1 . . . 103
13.2 Propagasi Balik sebagai Perhitungan Matriks Jacobian 105
13.3 Mengapa Suku $M(\boldsymbol{\theta})$ Kini Tidak Lagi Nol 107
13.4 *Generalized Gauss–Newton* dan Perlunya Regularisasi 108
13.5 Contoh Numerik: Mengapa Langkah Mentah Dapat Gagal . . . 109

14 MLP dengan Dua Lapisan Tersembunyi 113

14.1 Model MLP Dua Lapisan Tersembunyi 113
14.2 Propagasi Balik sebagai Rekursi Sinyal Galat 115
14.3 Peringkat Penuh Tidak Menjamin Kekondisian Baik 117
14.4 Contoh Numerik: Ketika Peringkat Penuh Saja Tidak Cukup . . 117
14.5 Eksperimen Publik: MLP pada Regresi Diabetes 120
14.6 MLP untuk Klasifikasi: Sigmoid dan Softmax 122
14.7 Eksperimen Publik: MLP Satu dan Dua Lapisan Tersembunyi pada Klasifikasi 122

15 Koneksi ke Teori Statistik: Matriks Informasi Fisher 127

15.1 Keluarga Eksponensial dan Model Linear Umum 127
15.2 Matriks Informasi Fisher: Definisi dan Identitas Dasarnya 128

15.3 Mengapa IRLS Merupakan *Fisher Scoring* 130

16 Konvergensi Global dengan Pencarian Garis Armijo–Wolfe 133

16.1 Mengapa Konvergensi Lokal Saja Tidak Cukup 133

16.2 Syarat Armijo dan Wolfe 134

16.3 Teorema Zoutendijk dan Jaminan Konvergensi Global 135

17 Ringkasan Sifat-Sifat Penting 139

17.1 Peta Keputusan untuk Memilih Metode 139

17.2 Kesimpulan Teoretis 140

17.3 Kesalahan yang Sering Terjadi 142

17.4 Prosedur Praktis dari Awal sampai Akhir 143

17.5 Penutup 143

18 Latihan Bertahap 145

18.1 Cara Menggunakan Latihan 145

18.2 Contoh Format Jawaban yang Baik 147

A Bukti Lengkap Teorema Konvergensi Lokal 149

B Bukti Teorema Zoutendijk 153

C Bukti Teorema Konvergensi Global Levenberg–Marquardt 155

D Glosarium 157

Daftar Pustaka 161

Tentang Penulis **165**

Daftar Gambar

1.1 Aliran perhitungan dari fitur dan parameter menuju prediksi serta residu 3

3.1 Ilustrasi langkah Newton pada kontur fungsi objektif di sekitar titik minimum 16

3.2 Model kuadrat lokal q di sekitar θ_k dan langkah Newton menuju minimum model 17

3.3 Persamaan normal sebagai proyeksi ortogonal ke ruang kolom Jacobian 24

4.1 Geometri SVD: lingkaran satuan dipetakan oleh J menjadi elips dengan setengah sumbu σ_1 dan σ_2 28

6.1 Perbandingan penurunan galat pada konvergensi linear dan kuadratik 40

7.1 Hubungan antara langkah Gauss–Newton, daerah kepercayaan, dan arah penurunan tercuram 46

7.2 Perbandingan strategi Marquardt dan Nielsen pada MLP 54

7.3 Lintasan *dogleg* $\mathbf{0} \to \boldsymbol{p}^C \to \boldsymbol{p}^{GN}$ dibandingkan dengan lintasan eksak $\boldsymbol{p}(\lambda)$ di dalam daerah kepercayaan $\|\boldsymbol{p}\| \le \Delta_k$. Kontur menyatakan model kuadrat $\phi(\boldsymbol{p})$ 57

8.1 Garis kuadrat terkecil tak berbobot dan berbobot pada data dengan ketidakpastian pengukuran yang berbeda 63

9.1 Kurva pembelajaran (selisih terhadap nilai minimum S^*) dan hasil pencocokan model eksponensial 71

9.2 Kurva pembelajaran dan pencocokan regresi kuadratik pada data Diabetes 74

10.1 Arsitektur perceptron satu lapisan untuk klasifikasi biner 76

10.2 Kiri: fungsi sigmoid dan turunannya. Kanan: besar gradien terhadap logit z untuk kerugian entropi-silang dan kerugian kuadrat ketika label benar $y = 1$ 77

10.3 Kurva sigmoid setelah satu iterasi IRLS pada contoh klasifikasi biner 84

11.1 Kurva konvergensi IRLS dan penurunan gradien pada data Titanic 89

12.1 Arsitektur perceptron satu lapisan dengan keluaran softmax multikelas 92

12.2 Kurva peluang tiga kelas setelah satu iterasi IRLS pada contoh softmax 100

12.3 Kurva pembelajaran dan akurasi latih pada data Iris 101

13.1 MLP dengan satu lapisan tersembunyi dan dua unit tersembunyi 104

13.2 Nilai objektif sepanjang arah Gauss–Newton dan arah Levenberg–Marquardt ($\lambda = 1$) pada MLP satu lapisan tersembunyi, beserta ramalan model linear untuk arah Gauss–Newton . . 111

14.1 MLP dengan dua lapisan tersembunyi dan dua unit pada setiap lapisan 114

14.2 Kurva LM dan hasil prediksi dua arsitektur MLP pada data Diabetes 121

14.3 Kurva LM dan akurasi uji MLP pada tugas biner dan multikelas 124

16.1 Syarat Armijo pada fungsi satu peubah $\phi(\alpha) = S(\boldsymbol{\theta}_k + \alpha \boldsymbol{p}_k)$ dan hasil pencarian garis mundur 135

Daftar Tabel

1.1 Peta studi kasus dan bukti hasil 4

6.1 Petunjuk praktis untuk membaca konvergensi 38

7.1 Perbandingan strategi Marquardt dan Nielsen 54

8.1 Contoh bobot dari simpangan baku pengukuran 62

9.1 Data numerik untuk regresi eksponensial 67

9.2 Hasil numerik regresi eksponensial 71

9.3 Hasil SLP untuk regresi Diabetes 74

10.1 Contoh data klasifikasi biner 82

11.1 Kecepatan konvergensi pada data Titanic 89

11.2 Matriks konfusi klasifikasi Titanic 90

11.3 Koefisien model logistik pada data Titanic 90

12.1 Contoh data klasifikasi multikelas 98

12.2 Hasil klasifikasi multikelas pada data Iris 102
12.3 Matriks konfusi IRLS pada data Iris 102

13.1 Dampak redaman dan lintasan LM pada MLP satu lapisan tersembunyi 110

14.1 Dampak redaman pada MLP dua lapisan tersembunyi 119
14.2 Lintasan nilai objektif MLP dua lapisan tersembunyi 119
14.3 Hasil MLP untuk regresi Diabetes 121
14.4 Hasil klasifikasi MLP satu dan dua lapisan tersembunyi 125

15.1 Korespondensi keluarga eksponensial dan variansi 130

17.1 Peta pemilihan metode berdasarkan masalah 140

Bab 1

Pendahuluan

Metode Gauss–Newton merupakan salah satu algoritma yang digunakan untuk menyelesaikan masalah kuadrat terkecil nonlinear (*nonlinear least squares*) (Madsen dkk., 2004; Nocedal dan Wright, 2006). Masalah ini muncul ketika suatu model perlu dicocokkan dengan data, tetapi hubungan antara parameter dan keluaran model bersifat nonlinear. Bab ini menguraikan permasalahan tersebut secara sistematis, mulai dari perumusan masalah hingga dasar matematis metode Gauss–Newton. Dengan demikian, setiap rumus yang disajikan pada bagian selanjutnya dapat ditelusuri berdasarkan kebutuhan matematis yang mendasarinya.

1.1 Rumusan Masalah

Sebelum berhadapan dengan matriks besar dan istilah teknis, kita mulai dari satu pertanyaan sederhana: bagaimana membuat model yang kita pilih sedekat mungkin dengan data yang kita amati? Seluruh buku ini pada dasarnya merupakan perjalanan untuk menjawab pertanyaan tersebut. Setiap konsep baru akan muncul karena diperlukan oleh langkah berikutnya.

Misalkan kita memiliki himpunan data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$ dan model parametrik $f(\boldsymbol{x}, \boldsymbol{\theta})$ dengan vektor parameter $\boldsymbol{\theta} \in \mathbb{R}^n$. Biasanya, jumlah parameter jauh lebih sedikit daripada jumlah data ($n \leq m$). Tujuan kita adalah membuat $f(\boldsymbol{x}_i, \boldsymbol{\theta})$

sedekat mungkin dengan y_i untuk setiap i. Selisih antara data dan prediksi model disebut **residu**:

$$r_i(\boldsymbol{\theta}) = y_i - f(\boldsymbol{x}_i, \boldsymbol{\theta}), \qquad i = 1, 2, \ldots, m. \tag{1.1}$$

Karena ada m residu sekaligus, kita menyusun semuanya menjadi satu vektor kolom agar lebih praktis:

$$\boldsymbol{r}(\boldsymbol{\theta}) = \begin{pmatrix} r_1(\boldsymbol{\theta}) \\ r_2(\boldsymbol{\theta}) \\ \vdots \\ r_m(\boldsymbol{\theta}) \end{pmatrix} \in \mathbb{R}^m. \tag{1.2}$$

Model yang baik membuat seluruh residu ini kecil secara bersamaan, bukan hanya sebagian. Namun, gagasan tersebut masih bersifat kualitatif. Kita perlu mengubahnya menjadi satu angka yang dapat diminimumkan. Caranya adalah menjumlahkan kuadrat semua residu. Kuadrat membuat kesalahan positif dan negatif sama-sama terhitung. Selain itu, fungsi yang dihasilkan juga mulus dan mudah diturunkan:

$$S(\boldsymbol{\theta}) = \frac{1}{2} \sum_{i=1}^{m} r_i(\boldsymbol{\theta})^2 = \frac{1}{2} \boldsymbol{r}(\boldsymbol{\theta})^\top \boldsymbol{r}(\boldsymbol{\theta}). \tag{1.3}$$

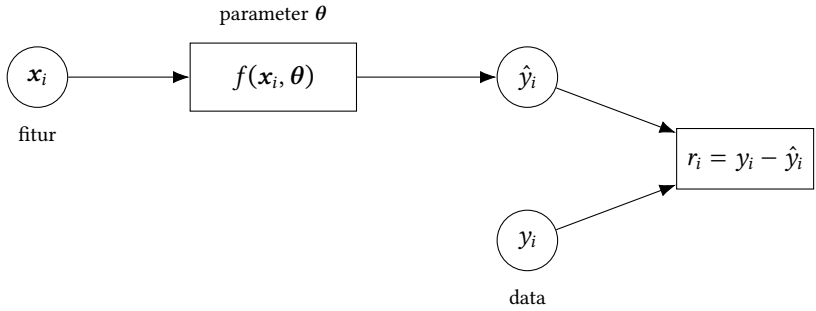
Kode MATLAB: Menghitung $S(\theta)$

```
clear; clc;
x = (0:3)';
y = [2.1; 3.9; 7.8; 15.2];
theta = [2.0; 0.6];
f = @(u,t) t(1)*exp(t(2)*u);
r = y - f(x, theta);
S = 0.5 * sum(r.^2);
disp(S)
```

Faktor $\frac{1}{2}$ di depan tidak mengubah letak titik minimum. Faktor ini hanya membuat turunan yang akan muncul terlihat lebih rapi. Dengan demikian,

tujuan kita jelas, yaitu mencari $\boldsymbol{\theta}^* \in \mathbb{R}^n$ yang meminimumkan $S(\boldsymbol{\theta})$. Dari tujuan yang sederhana inilah seluruh pembahasan berikutnya berangkat.

Sebelum masuk ke turunan matematis, mari kita lihat terlebih dahulu proses yang sedang dilakukan. Model $f(\boldsymbol{x}, \boldsymbol{\theta})$ sengaja dibiarkan abstrak sepanjang Bab 2–9. Model tersebut dapat berupa garis lurus atau jaringan saraf seperti yang dibahas pada Bab 13–14. Seluruh turunan Gauss–Newton hanya membutuhkan turunan model, bukan bentuk eksplisitnya. Aliran dari fitur menuju residu ditunjukkan pada Gambar 1.1.



Gambar 1.1: Aliran perhitungan dari fitur dan parameter menuju prediksi serta residu

Kotak $f(\boldsymbol{x}_i, \boldsymbol{\theta})$ pada diagram ini adalah "kotak hitam". Isinya akan dibuat konkret secara bertahap. Kita akan membahas fungsi eksponensial pada Bab 9, garis lurus dengan sigmoid pada Bab 10, beberapa garis dengan softmax pada Bab 12, serta jaringan berlapis pada Bab 13–14. Seluruh perangkat yang dibangun mulai Bab 2, seperti Jacobian, persamaan normal, dan analisis konvergensi, tetap berlaku apa pun isi kotak hitam tersebut.

1.2 Peta Studi Kasus dan Replikasi

Tidak semua materi membutuhkan eksperimen besar. Rumus dasar lebih mudah dipahami melalui data kecil yang dapat dihitung dengan tangan, sedangkan bagian algoritmik perlu diuji pada data yang lebih realistis. Karena itu, buku ini memakai dua jenis studi kasus: contoh kecil untuk memahami mekanisme dan

eksperimen publik untuk mengukur kinerja.

Tabel 1.1 merangkum hubungan antara materi, studi kasus, sumber data, dan bukti hasil yang akan kita periksa.

Tabel 1.1: Peta studi kasus dan bukti hasil

Materi	Studi kasus	Data	Bukti hasil
Bab 7	MLP kecil dan redaman LM	Delapan titik $y = \sin(x)$	Perbandingan S dan strategi λ
Bab 9	Regresi eksponensial	Empat titik, dihitung manual	Parameter, S, kurva pembelajaran
Subbab 9.2	SLP regresi Diabetes	442 baris, fitur BMI	Kurva GN/GD, RMSE uji
Bab 10–11	SLP biner Titanic	`train.csv`, 891 baris	Kurva kerugian, norma gradien, akurasi
Bab 12	SLP multikelas Iris	UCI Iris, 150 baris	Kurva pembelajaran, akurasi, matriks konfusi
Bab 13 dan 14.4	MLP satu dan dua lapisan	Delapan atau 16 titik $y = \sin(x)$	Langkah GN/LM, kondisi matriks, lintasan S
Subbab 14.5	MLP regresi Diabetes	442 baris, fitur BMI	Kurva LM, RMSE uji
Subbab 14.7	MLP biner dan multikelas	BCD dan Iris	Kurva LM, akurasi uji

Tabel 1.1 menunjukkan bahwa setiap studi kasus dirancang untuk mengilustrasikan konsep tertentu dan menghasilkan bukti yang dapat diperiksa. Agar proses memperoleh bukti tersebut dapat dipahami dan direplikasi, setiap studi kasus dilengkapi dengan bagian *Langkah Eksperimen* yang disajikan sebelum gambar atau tabel hasil. Bagian ini menjelaskan sumber data, pembagian data, persamaan model, parameter awal, aturan pembaruan, serta kriteria berhenti. Dengan susunan tersebut, kita akan memahami bagaimana hasil diperoleh dan dapat menelusuri setiap tahapan eksperimen secara sistematis. Kode program, apabila digunakan, berfungsi sebagai alat bantu replikasi, bukan sebagai satu-satunya sumber informasi mengenai prosedur eksperimen.

1.3 Pemanasan: Tiga Titik dan Garis Lurus

Sebelum notasi $\boldsymbol{r}(\boldsymbol{\theta}) \in \mathbb{R}^m$ dan $J(\boldsymbol{\theta}) \in \mathbb{R}^{m \times n}$ diperkenalkan pada Bab 2, kita coba terlebih dahulu kasus paling sederhana yang dapat dihitung dengan tangan.

Dengan cara ini, notasi umum tersebut akan terasa sebagai ringkasan dari sesuatu yang sudah kita kenal.

Misalkan terdapat $m = 3$ titik data, yaitu $(x_1, y_1) = (0, 1)$, $(x_2, y_2) = (1, 3)$, dan $(x_3, y_3) = (2, 4)$. Gunakan model garis lurus $f(x, \boldsymbol{\theta}) = \theta_1 + \theta_2 x$ dengan $\boldsymbol{\theta} = (\theta_1, \theta_2)$, yaitu intersep dan kemiringan. Ketiga residunya adalah

$$r_1 = 1 - \theta_1, \qquad r_2 = 3 - \theta_1 - \theta_2, \qquad r_3 = 4 - \theta_1 - 2\theta_2,$$

dan fungsi objektif (1.3) untuk kasus ini adalah polinomial kuadrat sederhana dalam dua peubah:

$$S(\theta_1, \theta_2) = \tfrac{1}{2}\Big[(1 - \theta_1)^2 + (3 - \theta_1 - \theta_2)^2 + (4 - \theta_1 - 2\theta_2)^2\Big].$$

Karena ini hanya fungsi dua peubah, kita dapat memakai kalkulus dasar. Titik minimum dicapai ketika kedua turunan parsial bernilai nol. Kita akan menurunkan S terhadap θ_1 dan θ_2 secara terpisah, lalu menunjukkan bagaimana kedua hasilnya berubah menjadi sistem persamaan linear. Untuk melihat asal kedua persamaan itu, gunakan aturan sederhana berikut. Jika g bergantung pada θ, maka

$$\frac{\partial}{\partial \theta}\left[\frac{1}{2} g(\theta)^2\right] = \frac{1}{2}\, 2 g(\theta) \frac{\partial g(\theta)}{\partial \theta} = g(\theta) \frac{\partial g(\theta)}{\partial \theta}.$$

Faktor 2 yang muncul ketika kuadrat diturunkan saling meniadakan dengan faktor $\frac{1}{2}$ di depan fungsi objektif. Turunan parsial terhadap θ_1 dihitung dengan menganggap θ_2 sebagai konstanta. Karena turunan setiap ekspresi di dalam kurung terhadap θ_1 sama dengan -1, diperoleh

$$\begin{aligned}
\frac{\partial S}{\partial \theta_1} &= \frac{1}{2}\Big[2(1 - \theta_1)(-1) + 2(3 - \theta_1 - \theta_2)(-1) + 2(4 - \theta_1 - 2\theta_2)(-1)\Big] \\
&= -(1 - \theta_1) - (3 - \theta_1 - \theta_2) - (4 - \theta_1 - 2\theta_2) \\
&= 3\theta_1 + 3\theta_2 - 8.
\end{aligned}$$

Untuk turunan terhadap θ_2, suku pertama tidak memberikan kontribusi kare-

na $1 - \theta_1$ tidak memuat θ_2. Turunan ekspresi kedua terhadap θ_2 adalah -1, sedangkan turunan ekspresi ketiga adalah -2. Jadi,

$$\begin{aligned}\frac{\partial S}{\partial \theta_2} &= \frac{1}{2}\Big[0 + 2(3 - \theta_1 - \theta_2)(-1) + 2(4 - \theta_1 - 2\theta_2)(-2)\Big] \\ &= -(3 - \theta_1 - \theta_2) - 2(4 - \theta_1 - 2\theta_2) \\ &= 3\theta_1 + 5\theta_2 - 11.\end{aligned}$$

Pada titik minimum, kedua turunan tersebut harus bernilai nol. Oleh sebab itu,

$$3\theta_1 + 3\theta_2 - 8 = 0, \qquad 3\theta_1 + 5\theta_2 - 11 = 0,$$

atau, setelah konstanta dipindahkan ke ruas kanan,

$$\begin{aligned}3\theta_1 + 3\theta_2 &= 8, \\ 3\theta_1 + 5\theta_2 &= 11.\end{aligned} \tag{1.4}$$

Inilah sistem persamaan linear yang harus diselesaikan. Kurangkan persamaan pertama dari persamaan kedua sehingga suku $3\theta_1$ saling menghapus:

$$(3\theta_1 + 5\theta_2) - (3\theta_1 + 3\theta_2) = 11 - 8 \quad \Longrightarrow \quad 2\theta_2 = 3.$$

Dengan demikian, $\theta_2 = 3/2 = 1.5$. Substitusikan nilai ini ke persamaan pertama pada (1.4):

$$3\theta_1 + 3\left(\frac{3}{2}\right) = 8 \quad \Longrightarrow \quad 3\theta_1 = \frac{7}{2} \quad \Longrightarrow \quad \theta_1 = \frac{7}{6} \approx 1.167.$$

Jadi, garis terbaik untuk ketiga titik tersebut adalah $f(x) = \frac{7}{6} + \frac{3}{2}x \approx 1.167 + 1.5x$. Prediksi pada $x = 0, 1, 2$ berturut-turut adalah $7/6$, $8/3$, dan $25/6$. Residu akhirnya ialah

$$\boldsymbol{r} = \left(-\frac{1}{6}, \frac{1}{3}, -\frac{1}{6}\right)^{\top}, \qquad S = \frac{1}{2}\left[\left(-\frac{1}{6}\right)^2 + \left(\frac{1}{3}\right)^2 + \left(-\frac{1}{6}\right)^2\right] = \frac{1}{12} \approx 0.0833.$$

Intuisi: Mengapa Titik Stasioner Ini Benar-Benar Minimum?

Syarat turunan pertama bernilai nol hanya memberi tahu bahwa kita telah menemukan titik stasioner. Untuk memastikan jenis titiknya, periksa matriks turunan kedua atau Hessian. Dengan menurunkan kedua hasil turunan pertama sekali lagi, kita memperoleh Hessian fungsi objektif

$$H = \begin{pmatrix} \dfrac{\partial^2 S}{\partial \theta_1^2} & \dfrac{\partial^2 S}{\partial \theta_1 \partial \theta_2} \\ \dfrac{\partial^2 S}{\partial \theta_2 \partial \theta_1} & \dfrac{\partial^2 S}{\partial \theta_2^2} \end{pmatrix} = \begin{pmatrix} 3 & 3 \\ 3 & 5 \end{pmatrix}.$$

Ambil sembarang vektor tak nol $\boldsymbol{v} = (v_1, v_2)^\top$. Bentuk kuadratiknya dapat ditulis sebagai

$$\boldsymbol{v}^\top H \boldsymbol{v} = 3v_1^2 + 6v_1 v_2 + 5v_2^2 = 3(v_1 + v_2)^2 + 2v_2^2 > 0.$$

Karena hasilnya positif untuk setiap $\boldsymbol{v} \neq \mathbf{0}$, Hessian H definit positif. Artinya, S berbentuk mangkuk yang melengkung ke atas. Dengan demikian, $\theta_1 = 7/6$ dan $\theta_2 = 3/2$ bukan hanya titik stasioner, melainkan minimum global yang tunggal.

Ada satu hal penting yang perlu diperhatikan. Karena f linear terhadap $\boldsymbol{\theta}$, masalah ini dapat diselesaikan langsung melalui dua persamaan linear. Iterasi tidak diperlukan. Jika X adalah matriks desain yang baris ke-i-nya $(1, x_i)$, kedua persamaan tersebut merupakan bentuk konkret dari **persamaan normal** $X^\top X\,\boldsymbol{\theta} = X^\top \boldsymbol{y}$. Pada Bab 2, Jacobian didefinisikan untuk residu sehingga $J = -X$ pada contoh garis lurus ini. Perbedaan tanda tersebut akan muncul pada ruas kanan persamaan untuk langkah koreksi $\boldsymbol{p}$. Ketika model tidak lagi linear, seperti model eksponensial pada Bab 9, barulah notasi Jacobian umum dan iterasi Gauss–Newton diperlukan.

Bab 2

Menuju Kondisi Optimal: Gradien dan Matriks Jacobian

Untuk mencari titik minimum sebuah fungsi yang mulus, langkah standar dalam kalkulus adalah mencari titik tempat gradiennya bernilai nol. Jadi, kita perlu menghitung $\nabla S(\boldsymbol{\theta})$ terlebih dahulu. Karena S dibangun dari m komponen vektor residu (1.2) sekaligus, kita mengumpulkan seluruh turunan parsial r_i terhadap seluruh θ_j ke dalam satu matriks agar lebih rapi. Matriks inilah yang disebut **matriks Jacobian** dan dilambangkan dengan $J(\boldsymbol{\theta}) \in \mathbb{R}^{m \times n}$:

$$J(\boldsymbol{\theta}) = \begin{pmatrix} \dfrac{\partial r_1}{\partial \theta_1} & \dfrac{\partial r_1}{\partial \theta_2} & \cdots & \dfrac{\partial r_1}{\partial \theta_n} \\ \dfrac{\partial r_2}{\partial \theta_1} & \dfrac{\partial r_2}{\partial \theta_2} & \cdots & \dfrac{\partial r_2}{\partial \theta_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial r_m}{\partial \theta_1} & \dfrac{\partial r_m}{\partial \theta_2} & \cdots & \dfrac{\partial r_m}{\partial \theta_n} \end{pmatrix} = \begin{bmatrix} \boldsymbol{j}_1 & \boldsymbol{j}_2 & \cdots & \boldsymbol{j}_n \end{bmatrix} \tag{2.1}$$

dengan $\boldsymbol{j}_l \in \mathbb{R}^m$ sebagai kolom ke-l, yaitu vektor sensitivitas seluruh residu terhadap satu parameter θ_l saja. Karena residu (1.1) berbentuk $r_i(\boldsymbol{\theta}) = y_i - f(\boldsymbol{x}_i, \boldsymbol{\theta})$ dan y_i tidak bergantung pada $\boldsymbol{\theta}$, setiap entri Jacobian dapat langsung

dihitung dari model:

$$[J(\boldsymbol{\theta})]_{il} = \frac{\partial r_i(\boldsymbol{\theta})}{\partial \theta_l} = -\frac{\partial f(\boldsymbol{x}_i, \boldsymbol{\theta})}{\partial \theta_l}, \qquad i = 1, \ldots, m, \quad l = 1, \ldots, n. \tag{2.2}$$

Intuisi

Jacobian adalah peta sensitivitas. Baris ke-i menjawab pertanyaan, "bagaimana residu pada data ke-i berubah ketika semua parameter diubah?" Kolom ke-l menjawab pertanyaan yang berbeda, yaitu "bagaimana semua residu berubah ketika hanya parameter θ_l diubah?" Karena residu didefinisikan sebagai data dikurangi prediksi, tanda turunannya berlawanan dengan tanda turunan model.

Contoh Angka

Misalkan $f(x, \boldsymbol{\theta}) = \theta_1 + \theta_2 x$ dan gunakan tiga data $(0, 1)$, $(1, 2)$, dan $(2, 4)$. Pada tebakan $\boldsymbol{\theta} = (1, 2)$, lakukan tiga langkah berikut.

1. Hitung residu. Prediksi model adalah $(1, 3, 5)$, sehingga $\boldsymbol{r} = (0, -1, -1)^\top$.
2. Turunkan residu terhadap parameter. Karena $r_i = y_i - \theta_1 - \theta_2 x_i$, diperoleh $\partial r_i / \partial \theta_1 = -1$ dan $\partial r_i / \partial \theta_2 = -x_i$.
3. Susun baris-barisnya. Dengan demikian,

$$J = \begin{pmatrix} -1 & 0 \\ -1 & -1 \\ -1 & -2 \end{pmatrix}.$$

Jadi, Jacobian bukan matriks misterius. Matriks ini hanyalah tabel yang mencatat pengaruh setiap parameter terhadap setiap residu.

Kode MATLAB: Membangun Matriks J secara Numerik dengan Beda Hingga

```
clear; clc;
x = (0:3)';
y = [2.1; 3.9; 7.8; 15.2];
theta = [2.0; 0.6];
f = @(u,t) t(1)*exp(t(2)*u);
n = length(theta); m = length(x);
J = zeros(m, n);
h = 1e-6;
for l = 1:n
    theta_plus = theta; theta_plus(l) = theta_plus(l) + h;
    r_plus = y - f(x, theta_plus);
    r_0    = y - f(x, theta);
    J(:,l) = (r_plus - r_0) / h;
end
disp(J)
```

Catatan Aljabar Linear

Perhatikan bahwa baris dan kolom J memiliki makna yang berbeda. Setiap **baris** ke-i berkaitan dengan satu titik data $(\boldsymbol{x}_i, y_i)$. Isinya adalah gradien residu r_i terhadap seluruh parameter, yaitu negatif dari gradien $f(\boldsymbol{x}_i, \boldsymbol{\theta})$. Setiap **kolom** ke-$l$ berkaitan dengan satu parameter θ_l. Isinya menunjukkan pengaruh perubahan kecil pada θ_l terhadap seluruh residu. Dalam bahasa aljabar linear, $J(\boldsymbol{\theta})$ merupakan representasi matriks dari pemetaan linear

$$T : \mathbb{R}^n \to \mathbb{R}^m, \qquad T(\boldsymbol{p}) = J(\boldsymbol{\theta})\boldsymbol{p},$$

Pemetaan tersebut mengambil perturbasi kecil pada parameter $\boldsymbol{p}$ di ruang *domain* berdimensi n. Selanjutnya, pemetaan itu menghasilkan perkiraan perubahan pada seluruh residu di ruang *kodomain* berdimensi m. Karena $n \leq m$, ruang domain lebih kecil daripada ruang kodomain. Fakta ini akan menentukan bentuk solusi yang dicari.

Sekarang, dengan $J(\boldsymbol{\theta})$ sudah di tangan, gradien $S(\boldsymbol{\theta})$ dapat dihitung memakai aturan rantai. Kita mulai dari turunan parsial terhadap satu komponen θ_j, kemudian merangkumnya dalam bentuk matriks. Fungsi $S(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{m} r_i(\boldsymbol{\theta})^2$

merupakan jumlah dari m suku kuadrat. Untuk satu suku, aturan rantai memberikan

$$\frac{d}{d\theta_j}\left[\frac{1}{2}u(\theta_j)^2\right] = u(\theta_j)\,u'(\theta_j).$$

Dengan menerapkan pola tersebut pada setiap residu, diperoleh

$$\frac{\partial S}{\partial \theta_j} = \sum_{i=1}^{m} r_i(\boldsymbol{\theta})\,\frac{\partial r_i(\boldsymbol{\theta})}{\partial \theta_j} = \sum_{i=1}^{m} r_i(\boldsymbol{\theta})\,[J(\boldsymbol{\theta})]_{ij}, \tag{2.3}$$

Dengan memakai definisi entri Jacobian $[J]_{ij} = \partial r_i/\partial \theta_j$ dari (2.2), jumlah pada (2.3) dapat ditulis sebagai

$$\sum_{i} r_i [J]_{ij} = (J^\top \boldsymbol{r})_j.$$

Kesetaraan ini berlaku untuk setiap $j = 1, \dots, n$. Karena itu, seluruh turunan parsial dapat dirangkum dalam satu persamaan vektor:

$$\nabla S(\boldsymbol{\theta}) = \sum_{i=1}^{m} r_i(\boldsymbol{\theta}) \nabla r_i(\boldsymbol{\theta}) = J(\boldsymbol{\theta})^\top \boldsymbol{r}(\boldsymbol{\theta}). \tag{2.4}$$

Intuisi

Perkalian $J^\top \boldsymbol{r}$ menggabungkan dua informasi. Vektor $\boldsymbol{r}$ memberi tahu seberapa besar kesalahan pada tiap data, sedangkan J memberi tahu parameter mana yang paling berpengaruh terhadap kesalahan tersebut. Gradien adalah hasil penggabungan keduanya. Jika suatu parameter memiliki pengaruh besar dan residu yang berkaitan dengannya juga besar, komponen gradiennya akan besar.

Contoh Angka

Gunakan J dan $\boldsymbol{r}$ dari contoh sebelumnya. Karena

$$J^\top = \begin{pmatrix} -1 & -1 & -1 \\ 0 & -1 & -2 \end{pmatrix}, \qquad \boldsymbol{r} = \begin{pmatrix} 0 \\ -1 \\ -1 \end{pmatrix},$$

maka

$$\nabla S = J^\top \boldsymbol{r} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}.$$

Artinya, pada tebakan tersebut fungsi objektif masih sensitif terhadap kedua parameter. Arah perbaikannya kemudian ditentukan dari gradien ini bersama informasi kelengkungan pada $J^\top J$.

Kode MATLAB: Gradien $\nabla S = J^\top r$

```
clear; clc;
x = (0:3)';
y = [2.1; 3.9; 7.8; 15.2];
theta = [2.0; 0.6];
e = exp(theta(2)*x);
r = y - theta(1)*e;
J = [-e, -theta(1)*x.*e];
grad = J' * r;
disp(grad)
```

Karena titik minimum $\boldsymbol{\theta}^*$ pasti membuat gradien bernilai nol, kita peroleh syarat perlunya:

$$J(\boldsymbol{\theta}^*)^\top \boldsymbol{r}(\boldsymbol{\theta}^*) = \mathbf{0}. \tag{2.5}$$

Persamaan (2.5) tetap *nonlinear* dalam $\boldsymbol{\theta}$ karena $\boldsymbol{r}$ dan J bergantung pada parameter tersebut. Oleh sebab itu, persamaan ini biasanya tidak dapat diselesaikan langsung dengan aljabar. Kita memerlukan pendekatan iteratif. Metode Newton adalah titik awal yang alami, sehingga dari sinilah perjalanan matematis kita berlanjut.

Bab 3

Dari Newton Klasik ke Aproksimasi Gauss–Newton

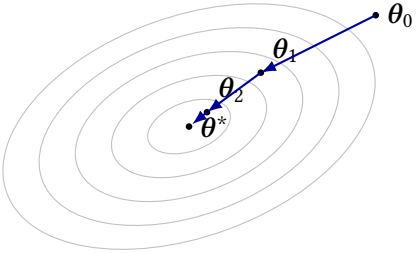
3.1 Metode Newton untuk Optimisasi

Analogi

Bayangkan kita berdiri di lereng bukit yang tertutup kabut dan sedang mencari titik terendah lembah. Kita dapat merasakan kemiringan tanah tepat di bawah kaki. Kemiringan ini adalah *gradien*. Kita juga dapat mengamati bagaimana kemiringan tersebut berubah di sekitar posisi kita. Perubahan ini berkaitan dengan *Hessian*, yaitu kelengkungan fungsi. Strategi yang hati-hati adalah bergerak sedikit demi sedikit ke arah turun, seperti pada penurunan gradien. Metode Newton mengambil langkah yang lebih berani. Metode ini menganggap daerah di sekitar posisi kita berbentuk *mangkuk parabola*, lalu menghitung titik terendah mangkuk tersebut. Jika perkiraannya akurat, metode ini dapat bergerak lebih cepat. Jika tidak, langkahnya dapat meleset, seperti yang akan kita lihat pada Subbab 13.5.

Gagasan metode Newton sederhana. Di sekitar iterasi saat ini $\boldsymbol{\theta}_k$, kita membuat aproksimasi kuadratik terhadap S. Kemudian, kita bergerak menuju titik minimum aproksimasi tersebut. Pada ruang parameter dua dimensi, kontur

$S(\boldsymbol{\theta})$ biasanya menyerupai elips di sekitar minimum $\boldsymbol{\theta}^*$. Pergerakan langkah Newton menuju solusi diperlihatkan pada Gambar 3.1.



Gambar 3.1: Ilustrasi langkah Newton pada kontur fungsi objektif di sekitar titik minimum

Setiap panah menunjukkan satu langkah $\boldsymbol{p}_k$. Pada titik $\boldsymbol{\theta}_k$, Newton menyamakan bentuk kontur di sekitarnya dengan elips sempurna, yaitu mangkuk kuadratik dari deret Taylor. Metode ini kemudian bergerak menuju pusat elips tersebut. Kontur fungsi yang sebenarnya biasanya tidak persis berbentuk elips. Karena itu, perkiraan Newton tidak selalu tepat dalam satu langkah. Namun, ketika sudah cukup dekat dengan $\boldsymbol{\theta}^*$, kontur semakin mendekati elips. Sifat lokal inilah yang menjadi dasar Teorema 1 pada Bab 6. Sekarang, mari kita turunkan langkah tersebut secara matematis. Ekspansi Taylor orde dua dari S di sekitar $\boldsymbol{\theta}_k$ memberikan

$$S(\boldsymbol{\theta}_k + \boldsymbol{p}) \approx \underbrace{S(\boldsymbol{\theta}_k) + \nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p} + \frac{1}{2}\boldsymbol{p}^\top H(\boldsymbol{\theta}_k)\boldsymbol{p}}_{=:\, q(\boldsymbol{p})}, \tag{3.1}$$

dengan $H(\boldsymbol{\theta}_k) = \nabla^2 S(\boldsymbol{\theta}_k)$ sebagai matriks Hessian, yaitu matriks turunan kedua. Ruas kanan (3.1) kita beri nama $q(\boldsymbol{p})$. Fungsi ini merupakan "mangkuk" pada diagram sebelumnya. Mencari titik minimumnya dapat dilakukan dengan kalkulus dasar. Turunkan q terhadap $\boldsymbol{p}$, lalu samakan hasilnya dengan nol. Suku pertama konstan terhadap $\boldsymbol{p}$. Suku kedua linear terhadap $\boldsymbol{p}$, sedangkan suku

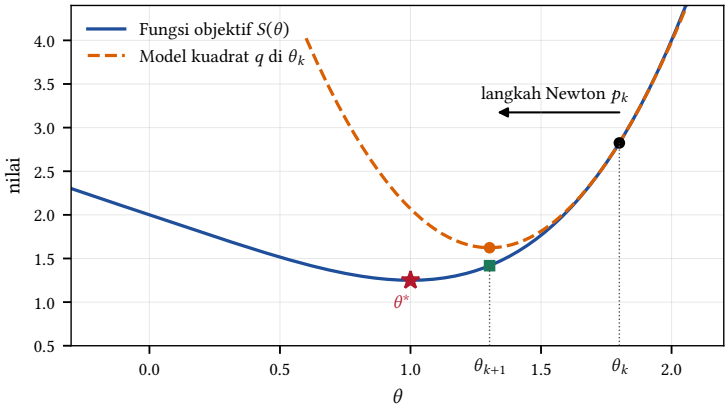
ketiga kuadratik. Dengan simetri H dan faktor $\frac{1}{2}$, turunannya menjadi

$$\nabla_{\boldsymbol{p}} q(\boldsymbol{p}) = \nabla S(\boldsymbol{\theta}_k) + H(\boldsymbol{\theta}_k)\boldsymbol{p}. \tag{3.2}$$

Menyamakan (3.2) dengan $\mathbf{0}$ dan menyelesaikan untuk $\boldsymbol{p}$ (mengalikan kedua ruas dengan $H(\boldsymbol{\theta}_k)^{-1}$ dari kiri, asalkan $H(\boldsymbol{\theta}_k)$ invertibel) memberikan arah langkah Newton:

$$\boldsymbol{p}_k = -H(\boldsymbol{\theta}_k)^{-1}\nabla S(\boldsymbol{\theta}_k). \tag{3.3}$$

Gambar 3.2 memperlihatkan gagasan ini dalam satu dimensi untuk fungsi $S(\theta) = \frac{1}{4}\theta^4 - \theta + 2$ dengan minimum $\theta^* = 1$. Di titik $\theta_k = 1.8$, model kuadrat q menyinggung S dan memiliki kelengkungan yang sama. Langkah (3.3) membawa iterasi ke minimum model, yaitu $\theta_{k+1} \approx 1.30$. Titik ini belum sama dengan θ^* karena S bukan fungsi kuadrat, tetapi jaraknya ke solusi sudah jauh lebih kecil.



Gambar 3.2: Model kuadrat lokal q di sekitar θ_k dan langkah Newton menuju minimum model

Kode MATLAB: Arah Langkah Newton

```
clear; clc;
H = [4, 1; 1, 3];
grad = [2; -1];
p = -H \ grad;
disp(p)
```

Untuk memakai rumus (3.3), kita perlu mengetahui bentuk eksplisit $H(\boldsymbol{\theta})$. Gradiennya sudah diketahui, yaitu $\nabla S(\boldsymbol{\theta}) = J(\boldsymbol{\theta})^{\top}\boldsymbol{r}(\boldsymbol{\theta})$ dari (2.4). Karena bentuk ini merupakan hasil kali dua besaran yang sama-sama bergantung pada $\boldsymbol{\theta}$, aturan hasil kali menghasilkan dua suku. Suku pertama berasal dari turunan $J(\boldsymbol{\theta})^{\top}$, sedangkan suku kedua berasal dari turunan $\boldsymbol{r}(\boldsymbol{\theta})$:

$$\nabla^2 S(\boldsymbol{\theta}) = \nabla\big[J(\boldsymbol{\theta})^{\top}\big]\,\boldsymbol{r}(\boldsymbol{\theta}) \;+\; J(\boldsymbol{\theta})^{\top}\nabla \boldsymbol{r}(\boldsymbol{\theta}). \tag{3.4}$$

Suku kedua pada (3.4) mudah dihitung. Karena $\nabla \boldsymbol{r}(\boldsymbol{\theta}) = J(\boldsymbol{\theta})^{\top}$, suku tersebut menjadi $J(\boldsymbol{\theta})^{\top}J(\boldsymbol{\theta})$. Inilah matriks Gram yang akan dipelajari berikutnya. Suku pertama lebih rumit karena melibatkan turunan kedua setiap r_i. Jika ditulis per komponen, entri ke-(k,l) suku tersebut adalah

$$\sum_{i=1}^{m} r_i(\boldsymbol{\theta})\frac{\partial^2 r_i}{\partial\theta_k\,\partial\theta_l}.$$

Gabungan kedua suku tersebut menghasilkan

$$H(\boldsymbol{\theta}) = J(\boldsymbol{\theta})^{\top}J(\boldsymbol{\theta}) + \sum_{i=1}^{m} r_i(\boldsymbol{\theta})\nabla^2 r_i(\boldsymbol{\theta}). \tag{3.5}$$

Suku kedua pada (3.5) merupakan bagian yang merepotkan. Suku ini membutuhkan turunan kedua setiap residu r_i, yaitu m matriks Hessian berukuran $n \times n$. Selain mahal dihitung, nilainya sering kali juga kecil. Jika model sudah cocok dengan data atau residunya hampir linear terhadap $\boldsymbol{\theta}$, suku tersebut dapat diabaikan tanpa banyak mengubah hasil.

3.2 Gagasan Gauss–Newton: Membuang Suku Kedua

Gagasan inti metode Gauss–Newton adalah mengabaikan suku kedua pada (3.5). Dengan demikian, Hessian didekati dengan

$$H(\boldsymbol{\theta}) \approx J(\boldsymbol{\theta})^{\top}J(\boldsymbol{\theta}). \tag{3.6}$$

Kode MATLAB: Aproksimasi Gauss–Newton

```
clear; clc;
J = [-1, 0; -1.8, -3.6; -3.3, -13.2; -6.0, -36.0];
H_approx = J' * J;
disp(H_approx)
```

Karena matriks J baru saja diperkenalkan, sekarang kita dapat melihat bentuk eksplisit $J^\top J$. Matriks inilah yang menggantikan peran Hessian pada setiap iterasi.

Transpos dari (2.1) adalah matriks $n \times m$ yang baris-barisnya merupakan kolom-kolom J yang ditulis sebagai vektor baris:

$$J(\boldsymbol{\theta})^\top = \begin{pmatrix} \boldsymbol{j}_1^\top \\ \boldsymbol{j}_2^\top \\ \vdots \\ \boldsymbol{j}_n^\top \end{pmatrix} \in \mathbb{R}^{n \times m}.$$

Perkalian $J^\top J$ dilakukan dengan aturan baris-kali-kolom. Baris ke-k dari $J^\top$ adalah $\boldsymbol{j}_k^\top$, sedangkan kolom ke-l dari J adalah $\boldsymbol{j}_l$. Jadi, entri ke-(k, l) merupakan **hasil kali dalam** (*inner product*) antara kolom ke-k dan kolom ke-l dari J:

$$\left[J(\boldsymbol{\theta})^\top J(\boldsymbol{\theta})\right]_{kl} = \boldsymbol{j}_k^\top \boldsymbol{j}_l = \sum_{i=1}^{m} \frac{\partial r_i}{\partial \theta_k} \cdot \frac{\partial r_i}{\partial \theta_l}, \qquad k, l = 1, \ldots, n. \tag{3.7}$$

Jika seluruh entri ini kita tuliskan, terbentuklah matriks $n \times n$ berikut:

$$J^\top J = \begin{pmatrix} \displaystyle\sum_{i=1}^{m} \left(\frac{\partial r_i}{\partial \theta_1}\right)^2 & \displaystyle\sum_{i=1}^{m} \frac{\partial r_i}{\partial \theta_1}\frac{\partial r_i}{\partial \theta_2} & \cdots & \displaystyle\sum_{i=1}^{m} \frac{\partial r_i}{\partial \theta_1}\frac{\partial r_i}{\partial \theta_n} \\ \displaystyle\sum_{i=1}^{m} \frac{\partial r_i}{\partial \theta_2}\frac{\partial r_i}{\partial \theta_1} & \displaystyle\sum_{i=1}^{m} \left(\frac{\partial r_i}{\partial \theta_2}\right)^2 & \cdots & \displaystyle\sum_{i=1}^{m} \frac{\partial r_i}{\partial \theta_2}\frac{\partial r_i}{\partial \theta_n} \\ \vdots & \vdots & \ddots & \vdots \\ \displaystyle\sum_{i=1}^{m} \frac{\partial r_i}{\partial \theta_n}\frac{\partial r_i}{\partial \theta_1} & \displaystyle\sum_{i=1}^{m} \frac{\partial r_i}{\partial \theta_n}\frac{\partial r_i}{\partial \theta_2} & \cdots & \displaystyle\sum_{i=1}^{m} \left(\frac{\partial r_i}{\partial \theta_n}\right)^2 \end{pmatrix} \tag{3.8}$$

Matriks yang entrinya berupa hasil kali dalam antarpasangan kolom suatu

matriks disebut **matriks Gram**. Ada dua sifat matriks Gram yang akan segera kita pakai untuk menjamin bahwa metode ini masuk akal secara matematis. Karena definisinya baru saja kita bahas, mari kita buktikan kedua sifat itu sekarang.

Catatan Aljabar Linear: Mengapa $J^\top J$ Selalu Simetris dan Semidefinit Positif

(a) Simetri. Ingat sifat dasar transpos perkalian matriks, $(AB)^\top = B^\top A^\top$. Dengan sifat tersebut,

$$\left(J^\top J\right)^\top = J^\top \left(J^\top\right)^\top = J^\top J.$$

Hal yang sama juga tampak langsung dari (3.7). Kita memiliki $[J^\top J]_{kl} = \boldsymbol{j}_k^\top \boldsymbol{j}_l = \boldsymbol{j}_l^\top \boldsymbol{j}_k = [J^\top J]_{lk}$ karena hasil kali dalam bersifat komutatif. Sifat simetri ini menjamin bahwa seluruh nilai eigen $J^\top J$ bernilai real, sesuai Teorema Spektral. Sifat tersebut akan berguna ketika kita membahas nilai singular di Bab 4.

(b) Semidefinit positif. Ambil sembarang $\boldsymbol{x} \in \mathbb{R}^n$, lalu tinjau bentuk kuadratik berikut:

$$\boldsymbol{x}^\top \left(J^\top J\right) \boldsymbol{x} = (J\boldsymbol{x})^\top (J\boldsymbol{x}) = \|J\boldsymbol{x}\|_2^2 \geq 0.$$

Karena norma kuadrat tidak pernah negatif, $J^\top J$ semidefinit positif untuk sembarang J. Tidak diperlukan syarat tambahan. Selain itu, kesamaan $\boldsymbol{x}^\top (J^\top J)\boldsymbol{x} = 0$ hanya terjadi jika $J\boldsymbol{x} = \boldsymbol{0}$. Artinya, $\boldsymbol{x}$ berada di ruang nol (*kernel*) J. Dari sini kita memperoleh kesetaraan penting:

$$\begin{aligned} J^\top J \text{ definit positif (tegas)} &\iff \ker(J) = \{\boldsymbol{0}\} \\ &\iff J \text{ berperingkat penuh kolom.} \end{aligned}$$

Kesetaraan inilah yang menjamin $\operatorname{rank}(J^\top J) = \operatorname{rank}(J)$, karena keduanya berbagi kernel yang sama persis.

Poin (b) di atas penting untuk digarisbawahi. Matriks $J^\top J$ dapat dibalik

(*invertible*) tepat ketika kolom-kolom J bebas linear. Syarat ini akan digunakan sampai akhir buku.

3.3 Menurunkan Persamaan Normal Gauss–Newton

Ada cara lain yang lebih intuitif untuk menurunkan aproksimasi $H \approx J^\top J$ pada (3.6). Cara ini tidak menggunakan Hessian, tetapi langsung melinearkan model. Pertimbangkan ekspansi Taylor orde pertama residu $\boldsymbol{r}$ di sekitar $\boldsymbol{\theta}_k$:

$$\boldsymbol{r}(\boldsymbol{\theta}_k + \boldsymbol{p}) \approx \boldsymbol{r}(\boldsymbol{\theta}_k) + J(\boldsymbol{\theta}_k)\boldsymbol{p}. \tag{3.9}$$

Jika aproksimasi linear ini disubstitusikan ke $S(\boldsymbol{\theta})$ pada (1.3), masalah minimisasi nonlinear berubah menjadi masalah kuadrat terkecil *linear* yang jauh lebih mudah. Masalah linear ini diselesaikan ulang pada setiap iterasi:

$$\boldsymbol{p}_k = \arg\min_{\boldsymbol{p}\in\mathbb{R}^n} \; \phi(\boldsymbol{p}), \qquad \phi(\boldsymbol{p}) := \frac{1}{2}\left\|\boldsymbol{r}_k + J_k\boldsymbol{p}\right\|_2^2, \tag{3.10}$$

dengan notasi singkat $\boldsymbol{r}_k := \boldsymbol{r}(\boldsymbol{\theta}_k)$ dan $J_k := J(\boldsymbol{\theta}_k)$. Mari kita uraikan norma kuadrat pada ϕ:

$$\phi(\boldsymbol{p}) = \frac{1}{2}(\boldsymbol{r}_k + J_k\boldsymbol{p})^\top(\boldsymbol{r}_k + J_k\boldsymbol{p}) = \frac{1}{2}\boldsymbol{r}_k^\top\boldsymbol{r}_k + \boldsymbol{p}^\top J_k^\top\boldsymbol{r}_k + \frac{1}{2}\boldsymbol{p}^\top J_k^\top J_k\boldsymbol{p}. \tag{3.11}$$

Matriks $J_k^\top J_k$ kembali muncul secara alami. Menurut (3.11), fungsi ϕ berbentuk kuadratik. Jika J_k berperingkat penuh kolom, fungsi ini konveks, sehingga syarat $\nabla_{\boldsymbol{p}}\phi(\boldsymbol{p}) = \boldsymbol{0}$ cukup dan perlu untuk mencapai minimum.

Jika kita belum terbiasa menurunkan fungsi terhadap vektor, kita dapat memeriksa setiap komponennya satu per satu. Misalkan $\boldsymbol{p} = (p_1, \dots, p_n)^\top$ dan $A := J_k^\top J_k$. Karena A simetris, berlaku

$$\frac{\partial}{\partial p_j}\left(\boldsymbol{p}^\top J_k^\top\boldsymbol{r}_k\right) = \left[J_k^\top\boldsymbol{r}_k\right]_j, \qquad \frac{\partial}{\partial p_j}\left(\frac{1}{2}\boldsymbol{p}^\top A\boldsymbol{p}\right) = \sum_{l=1}^{n} A_{jl}p_l = [A\boldsymbol{p}]_j.$$

Persamaan kedua merupakan bentuk banyak-peubah dari turunan $\frac{1}{2}ap^2 = ap$.

Jika hasil untuk $j = 1, \dots, n$ ditumpuk kembali menjadi sebuah vektor, diperoleh

$$\nabla_{\boldsymbol{p}} \phi(\boldsymbol{p}) = J_k^{\top} \boldsymbol{r}_k + J_k^{\top} J_k \boldsymbol{p} = \boldsymbol{0}. \tag{3.12}$$

Menata ulang (3.12) menghasilkan **persamaan normal Gauss–Newton**:

$$\boxed{J_k^{\top} J_k \, \boldsymbol{p}_k = -J_k^{\top} \boldsymbol{r}_k} \tag{3.13}$$

Jika J_k berperingkat penuh kolom, maka $J_k^{\top} J_k$ definit positif dan dapat dibalik. Oleh karena itu, arah langkah dapat dituliskan secara eksplisit:

$$\boldsymbol{p}_k = -\left(J_k^{\top} J_k\right)^{-1} J_k^{\top} \boldsymbol{r}_k. \tag{3.14}$$

Contoh Angka

Berikut ini adalah contoh penyelesaian persamaan normal secara lengkap. Ambil

$$J = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}, \qquad \boldsymbol{r} = \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}.$$

Langkah pertama adalah menghitung

$$J^{\top} J = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}, \qquad J^{\top} \boldsymbol{r} = \begin{pmatrix} 2 \\ -1 \end{pmatrix}.$$

Langkah kedua adalah menyelesaikan sistem

$$\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \boldsymbol{p} = -\begin{pmatrix} 2 \\ -1 \end{pmatrix}.$$

Eliminasi sederhana menghasilkan $\boldsymbol{p} = (-5/3,\ 4/3)^{\top}$. Jadi, persamaan normal mengubah residu awal menjadi usulan perubahan pada dua parameter.

Kode MATLAB: Menyelesaikan Persamaan Normal

```
clear; clc;
J = [-1, 0; -1.8, -3.6; -3.3, -13.2; -6.0, -36.0];
r = [0.1; 0.25; 1.1; 3.0];
p = -(J'*J) \ (J'*r);
disp(p)
```

Sebelum melanjutkan ke algoritma lengkap, kita perlu memahami makna geometris persamaan normal ini. Makna tersebut memberikan salah satu intuisi terkuat dalam metode Gauss–Newton.

Catatan Aljabar Linear: Persamaan Normal sebagai Proyeksi Ortogonal

Subproblem (3.10) pada dasarnya mencari $\boldsymbol{p}$ yang membuat $J_k\boldsymbol{p}$ sedekat mungkin (dalam norma Euklides) dengan $-\boldsymbol{r}_k$. Untuk $\boldsymbol{p}$ berapa pun, vektor $J_k\boldsymbol{p}$ selalu berada di dalam **ruang kolom** $\mathrm{Col}(J_k) \subseteq \mathbb{R}^m$, yaitu himpunan seluruh kombinasi linear dari kolom-kolom J_k. Karena itu, titik di $\mathrm{Col}(J_k)$ yang paling dekat dengan vektor $-\boldsymbol{r}_k$ tidak lain adalah **proyeksi ortogonal** dari $-\boldsymbol{r}_k$ ke $\mathrm{Col}(J_k)$:

$$J_k\boldsymbol{p}_k = \mathrm{Proj}_{\mathrm{Col}(J_k)}(-\boldsymbol{r}_k).$$

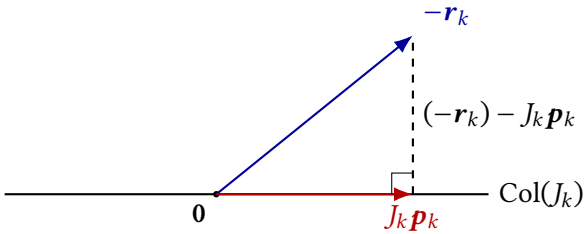
Ciri khas proyeksi ortogonal adalah sisa proyeksinya, yaitu $(-\boldsymbol{r}_k) - J_k\boldsymbol{p}_k$, tegak lurus terhadap seluruh $\mathrm{Col}(J_k)$. Dengan kata lain, sisa tersebut tegak lurus terhadap setiap kolom J_k:

$$\begin{aligned}\boldsymbol{j}_l^\top\left[(-\boldsymbol{r}_k) - J_k\boldsymbol{p}_k\right] = 0 \ \ \forall l \iff J_k^\top\left(-\boldsymbol{r}_k - J_k\boldsymbol{p}_k\right) &= \boldsymbol{0} \\ \iff J_k^\top J_k\boldsymbol{p}_k &= -J_k^\top\boldsymbol{r}_k,\end{aligned}$$

yang persis sama dengan persamaan normal (3.13). Jadi, **setiap iterasi Gauss–Newton pada dasarnya merupakan proyeksi ortogonal**. Arah koreksi ideal $-\boldsymbol{r}_k$ diproyeksikan ke ruang kolom Jacobian. Ruang ini memuat seluruh arah perubahan residu yang dapat dicapai secara linear melalui perubahan parameter.

Gambar 3.3 meringkas gagasan ini secara visual. Vektor $-\boldsymbol{r}_k$ (arah koreksi

yang idealnya kita inginkan) tidak selalu dapat dicapai persis oleh $J_k\boldsymbol{p}$, karena $J_k\boldsymbol{p}$ terkurung di dalam ruang kolom $\mathrm{Col}(J_k)$. Yang terbaik yang dapat kita lakukan adalah memproyeksikan $-\boldsymbol{r}_k$ secara tegak lurus ke ruang tersebut.



Gambar 3.3: Persamaan normal sebagai proyeksi ortogonal ke ruang kolom Jacobian

Garis putus-putus adalah sisa proyeksi. Sudut siku-siku kecil di kakinya menandakan bahwa sisa tersebut ortogonal terhadap $\mathrm{Col}(J_k)$. Kondisi ini sama dengan syarat yang kita turunkan secara aljabar.

3.4 Algoritma Iteratif

Dengan arah langkah $\boldsymbol{p}_k$ sudah di tangan, iterasi Gauss–Newton menjadi

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha_k \boldsymbol{p}_k = \boldsymbol{\theta}_k - \alpha_k \left(J_k^\top J_k\right)^{-1} J_k^\top \boldsymbol{r}_k, \tag{3.15}$$

dengan $\alpha_k \in (0, 1]$ sebagai parameter panjang langkah (*step length*). Nilainya dapat ditentukan melalui pencarian garis agar S terjamin menurun, atau cukup ditetapkan $\alpha_k = 1$ pada versi paling sederhana yang biasa disebut “Gauss–Newton murni”.

Sebagai pembanding, penurunan gradien hanya memakai gradien pada (2.4). Dengan laju belajar tetap $\eta_k > 0$, pembaruannya adalah

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta_k \nabla S(\boldsymbol{\theta}_k) = \boldsymbol{\theta}_k - \eta_k J_k^\top \boldsymbol{r}_k. \tag{3.16}$$

Kode MATLAB: Iterasi Gauss–Newton Lengkap

```
clear; clc;
x = (0:3)';
y = [2.1; 3.9; 7.8; 15.2];
theta0 = [2.0; 0.6];
max_iter = 20;
tol = 1e-8;
f = @(u,t) t(1)*exp(t(2)*u);
jacobian = @(u,t) [-exp(t(2)*u), ...
    -t(1)*u.*exp(t(2)*u)];
theta = theta0;
for k = 1:max_iter
    r = y - f(x, theta);
    J = jacobian(x, theta);
    p = -(J'*J) \ (J'*r);
    theta = theta + p;
    if norm(J'*r) < tol
        break;
    end
end
disp(theta)
```

Algoritma: Metode Gauss–Newton

1. Berikan tebakan awal $\boldsymbol{\theta}_0$, toleransi $\varepsilon > 0$, dan set $k = 0$.
2. Hitung residu $\boldsymbol{r}_k = \boldsymbol{r}(\boldsymbol{\theta}_k)$ dan Jacobian $J_k = J(\boldsymbol{\theta}_k)$.
3. Selesaikan sistem linear persamaan normal $J_k^\top J_k \boldsymbol{p}_k = -J_k^\top \boldsymbol{r}_k$ (lihat Bab 4 untuk cara menyelesaikannya secara stabil).
4. Tentukan panjang langkah α_k (misalnya melalui *backtracking line search*).
5. Perbarui parameter menurut (3.15): $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha_k \boldsymbol{p}_k$.
6. Jika $\|J_k^\top \boldsymbol{r}_k\| < \varepsilon$ atau $\|\boldsymbol{p}_k\| < \varepsilon$, berhenti. Jika belum, $k \leftarrow k + 1$ dan kembali ke langkah 2.

Langkah 3 pada algoritma di atas menyembunyikan pertanyaan praktis yang penting. Bagaimana sistem linear $J_k^\top J_k \boldsymbol{p}_k = -J_k^\top \boldsymbol{r}_k$ diselesaikan di komputer? Cara yang tampak paling langsung adalah membentuk $J_k^\top J_k$ lalu membalikkannya. Namun, cara tersebut justru berbahaya secara numerik. Berikut ini kita menelusuri alasannya sekaligus memperkenalkan konsep nilai singular yang akan digunakan untuk menganalisis kecepatan konvergensi.

Bab 4

Menyelesaikan Persamaan Normal secara Stabil

4.1 Dekomposisi Nilai Singular (SVD) dan Bilangan Kondisi

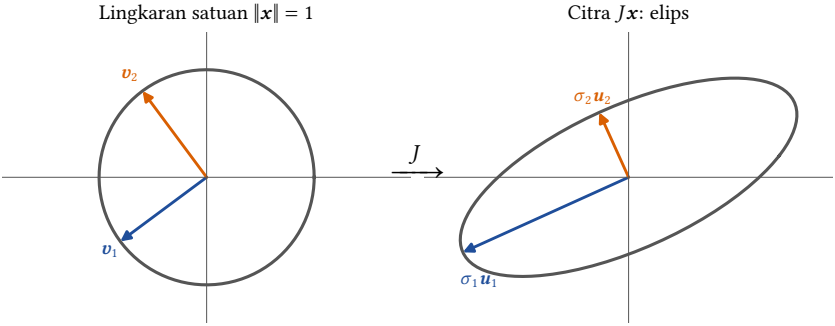
Setiap matriks $J_k \in \mathbb{R}^{m \times n}$ (dengan $m \geq n$) selalu dapat difaktorkan sebagai

$$J_k = U \Sigma V^\top, \tag{4.1}$$

dengan $U \in \mathbb{R}^{m \times n}$ dan $V \in \mathbb{R}^{n \times n}$ yang memiliki kolom-kolom ortonormal. Matriks $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$ memuat **nilai singular** $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$ dari J_k. Secara geometris, nilai singular mengukur seberapa besar J_k meregangkan sebuah vektor pada setiap arah utama. Faktorisasi ini disebut **dekomposisi nilai singular** atau SVD (*singular value decomposition*) (Golub dan Van Loan, 2013; Trefethen dan Bau, 1997). Matriks J_k berperingkat penuh kolom tepat ketika seluruh $\sigma_i > 0$. Karena itu, $\sigma_n = \sigma_{\min}(J_k)$ dapat dipakai untuk mengukur kedekatan J_k terhadap keadaan singular. Nilai $\sigma_{\min}$ yang semakin kecil menandakan bahwa kehilangan peringkat semakin dekat.

Makna geometris SVD diperlihatkan pada Gambar 4.1 untuk sebuah matriks 2×2. Matriks tersebut memetakan lingkaran satuan menjadi elips. Arah $\boldsymbol{v}_1$

dan $\boldsymbol{v}_2$ dipetakan ke sumbu-sumbu elips $\sigma_1 \boldsymbol{u}_1$ dan $\sigma_2 \boldsymbol{u}_2$. Semakin pipih elips tersebut, semakin besar rasio σ_1/σ_2 dan semakin sulit sistem linear diselesaikan secara akurat. Rasio ini akan kita sebut bilangan kondisi.



Gambar 4.1: Geometri SVD: lingkaran satuan dipetakan oleh J menjadi elips dengan setengah sumbu σ_1 dan σ_2

Kode MATLAB: SVD dan Bilangan Kondisi

```
clear; clc;
J = [1, 0; 0, 2; 1, 1];
[U, Sigma, V] = svd(J, 'econ');
sigma_vals = diag(Sigma);
sigma_min = sigma_vals(end);
kappa2_J  = cond(J);
disp(sigma_min)
disp(kappa2_J)
```

Dengan SVD ini di tangan, mari kita hitung ulang matriks Gram tadi:

$$J_k^\top J_k = V \Sigma U^\top U \Sigma V^\top = V \Sigma^2 V^\top, \tag{4.2}$$

dengan memakai fakta $U^\top U = I$ karena kolom U ortonormal. Matriks $J_k^\top J_k$ dan J_k memiliki matriks V yang sama. Karena itu, menurut (4.2), nilai eigen $J_k^\top J_k$ adalah σ_i^2. **Bilangan kondisi** mengukur sensitivitas solusi sistem linear terhadap galat pembulatan. Bilangan ini didefinisikan sebagai rasio nilai singular terbesar terhadap nilai singular terkecil, yaitu $\kappa_2(A) = \sigma_{\max}(A)/\sigma_{\min}(A)$. Untuk

matriks Gram, rasionya menjadi

$$\kappa_2\left(J_k^\top J_k\right) = \frac{\sigma_1^2}{\sigma_n^2} = \left[\kappa_2(J_k)\right]^2 . \tag{4.3}$$

Contoh Angka

Ambil contoh sederhana

$$J = \begin{pmatrix} 1 & 0 \\ 0 & 2 \\ 0 & 0 \end{pmatrix}.$$

Matriks ini meregangkan arah pertama sebesar 1 dan arah kedua sebesar 2. Jadi, nilai singularnya adalah $\sigma_1 = 2$ dan $\sigma_2 = 1$. Bilangan kondisi J adalah 2. Jika kita membentuk matriks Gram,

$$J^\top J = \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix},$$

bilangan kondisinya menjadi 4. Contoh kecil ini memperlihatkan mengapa pembentukan $J^\top J$ dapat memperburuk masalah numerik.

Persamaan (4.3) merupakan peringatan penting. **Membentuk $J_k^\top J_k$ secara eksplisit akan mengkuadratkan bilangan kondisinya**. Misalkan J_k sudah berkondisi buruk, dengan $\kappa_2(J_k) \approx 10^6$. Dalam kasus ini, $\kappa_2(J_k^\top J_k) \approx 10^{12}$. Galat pembulatan kemudian dapat mendominasi solusi $\boldsymbol{p}_k$ yang dihitung dari persamaan normal secara naif.

4.2 Alternatif Stabil: Dekomposisi QR

Untungnya, subproblem (3.10) dapat diselesaikan tanpa pernah membentuk $J_k^\top J_k$ sama sekali. Misalkan dekomposisi QR ekonomis (*thin QR*) dari J_k adalah

$$J_k = Q_k R_k, \qquad Q_k \in \mathbb{R}^{m\times n} \text{ ortonormal}, \quad R_k \in \mathbb{R}^{n\times n} \text{ segitiga atas}. \tag{4.4}$$

Substitusikan (4.4) ke persamaan normal (3.13):

$$J_k^\top J_k \boldsymbol{p}_k = R_k^\top Q_k^\top Q_k R_k \boldsymbol{p}_k = R_k^\top R_k \boldsymbol{p}_k = -J_k^\top \boldsymbol{r}_k = -R_k^\top Q_k^\top \boldsymbol{r}_k. \tag{4.5}$$

Karena R_k invertibel (selama J_k berperingkat penuh kolom), kedua ruas (4.5) tinggal dikalikan dengan $(R_k^\top)^{-1}$ dari kiri, dan kita peroleh sistem segitiga yang jauh lebih ramah secara numerik:

$$\boxed{R_k \, \boldsymbol{p}_k = -Q_k^\top \boldsymbol{r}_k} \tag{4.6}$$

Intuisi

QR mengganti satu sistem yang berpotensi buruk dengan dua langkah yang lebih aman. Pertama, $Q_k^\top$ memproyeksikan residu ke ruang kolom Jacobian. Kedua, sistem segitiga (4.6) diselesaikan dengan substitusi mundur. Kita tidak perlu membentuk $J_k^\top J_k$.

Contoh Angka

Untuk matriks pada contoh SVD sebelumnya, kolom-kolomnya sudah saling tegak lurus. Karena itu,

$$Q = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}, \qquad R = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}.$$

Penyelesaian dilakukan melalui $R\boldsymbol{p} = -Q^\top \boldsymbol{r}$. Perhatikan bahwa kita hanya menyelesaikan sistem dengan matriks segitiga R, bukan dengan matriks Gram.

Kode MATLAB: Menyelesaikan Langkah melalui QR

```
clear; clc;
J = [1, 0; 0, 2; 1, 1];
r = [1; -1; 0.5];
[Q, R] = qr(J, 0);
```

Kode MATLAB: Menyelesaikan Langkah melalui QR *(lanjutan)*

```
p = R \ (-Q'*r);
p_alt = -J \ r;
disp([p, p_alt])
```

Sistem ini diselesaikan melalui substitusi mundur dengan kompleksitas $O(n^2)$. Pendekatan ini sama sekali **tidak** mengkuadratkan bilangan kondisi J_k. Inilah pendekatan yang dipakai pustaka numerik standar, misalnya `lstsq` pada LAPACK (Björck, 1996; Golub dan Van Loan, 2013).

4.3 Kasus Kekurangan Peringkat: Solusi melalui SVD

Kadang-kadang J_k tidak berperingkat penuh kolom. Hal ini dapat terjadi jika model memiliki parameter berlebih yang pengaruhnya terhadap data tidak dapat dibedakan. Dalam situasi tersebut, $J_k^\top J_k$ menjadi singular sehingga (3.13) memiliki banyak solusi. Solusi dengan norma minimum dapat diperoleh langsung dari SVD (4.1) melalui **pseudoinvers Moore–Penrose**:

$$\boldsymbol{p}_k = -J_k^{+}\boldsymbol{r}_k = -V\Sigma^{+}U^\top \boldsymbol{r}_k, \qquad \Sigma^{+} = \operatorname{diag}\left(\frac{1}{\sigma_1}, \dots, \frac{1}{\sigma_r}, 0, \dots, 0\right), \tag{4.7}$$

dengan $r = \operatorname{rank}(J_k) < n$. Nilai singular yang secara numerik nol, yaitu yang berada di bawah ambang tertentu, diperlakukan sebagai nol pada Σ^{+} dalam (4.7). Langkah ini secara efektif meregularisasi arah $\boldsymbol{p}$ yang tidak dapat diidentifikasi dari data.

Kode MATLAB: Solusi Kekurangan Peringkat melalui Pseudoinvers

```
clear; clc;
J = [1, 2; 2, 4; 3, 6];
r = [1; -1; 0.5];
p = -pinv(J) * r;
disp(p)
```

Dengan bekal SVD dan konsep $\sigma_{\min}$ ini, kita sekarang siap menjawab dua pertanyaan yang lebih mendasar: apakah arah $\boldsymbol{p}_k$ selalu membuat kemajuan

(Bab 5), dan seberapa cepat iterasi ini konvergen (Bab 6).

Bab 5

Menjamin Kemajuan Setiap Langkah: Arah Turun

Sebelum menanyakan seberapa *cepat* metode ini konvergen, kita perlu memastikan bahwa langkahnya bergerak ke arah yang benar. Dengan kata lain, apakah arah $\boldsymbol{p}_k$ benar-benar mengurangi nilai $S(\boldsymbol{\theta})$, atau justru membuatnya bertambah?

5.1 Apa yang Dimaksud dengan Arah Turun?

Ambil sebuah titik saat ini $\boldsymbol{\theta}_k$ dan sebuah arah $\boldsymbol{p}$. Jika kita bergerak sedikit sejauh $\alpha > 0$, titik baru tersebut adalah

$$\boldsymbol{\theta}(\alpha) = \boldsymbol{\theta}_k + \alpha \boldsymbol{p}. \tag{5.1}$$

Untuk mengetahui akibat gerakan itu, kita tidak perlu langsung menghitung nilai fungsi pada banyak titik. Cukup periksa turunan S sepanjang garis (5.1) terhadap α ketika $\alpha = 0$:

$$\frac{d}{d\alpha} S(\boldsymbol{\theta}_k + \alpha \boldsymbol{p})\bigg|_{\alpha=0} = \nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}. \tag{5.2}$$

Jika nilai pada (5.2) negatif, maka S pasti turun untuk langkah yang cukup kecil. Inilah arti arah turun. Perhatikan bahwa istilah ini tidak mengatakan bahwa setiap panjang langkah aman. Istilah ini hanya memberi informasi tentang arah gerakan pada titik awal.

Intuisi

Bayangkan fungsi objektif sebagai permukaan bukit. Gradien menunjuk ke arah kenaikan paling curam. Karena itu, $-\nabla S$ menunjuk ke arah penurunan paling curam. Gauss–Newton tidak selalu memilih arah paling curam. Metode ini memilih arah yang sudah dikoreksi oleh bentuk lokal model melalui $J^\top J$. Koreksi ini biasanya membuat gerakan lebih efisien, tetapi syarat arah turunnya tetap perlu diperiksa.

5.2 Arah yang Benar Belum Tentu Berarti Langkah Penuh

Kesalahan yang sering terjadi adalah menganggap $\boldsymbol{p}_k$ aman dipakai dengan $\alpha_k = 1$ pada semua keadaan. Padahal, model kuadrat yang dipakai untuk memperoleh langkah hanya merupakan pendekatan lokal. Untuk melihat persoalannya, tuliskan ekspansi Taylor satu dimensi:

$$S(\boldsymbol{\theta}_k + \alpha \boldsymbol{p}_k) = S(\boldsymbol{\theta}_k) + \alpha \nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k + \frac{\alpha^2}{2} \boldsymbol{p}_k^\top H(\widetilde{\boldsymbol{\theta}}) \boldsymbol{p}_k, \tag{5.3}$$

dengan $\widetilde{\boldsymbol{\theta}}$ berada di antara $\boldsymbol{\theta}_k$ dan $\boldsymbol{\theta}_k + \alpha \boldsymbol{p}_k$. Pada (5.3), suku pertama setelah $S(\boldsymbol{\theta}_k)$ menentukan arah awal. Suku kuadrat menentukan apakah langkah tersebut terlalu jauh.

Untuk itu, algoritma praktis memisahkan dua keputusan. Pertama, hitung arah menggunakan Gauss–Newton. Kedua, pilih panjang langkah menggunakan pencarian garis. Jika percobaan dengan $\alpha = 1$ menaikkan S, nilai α diperkecil, misalnya menjadi $1/2$, $1/4$, dan seterusnya, sampai penurunan yang cukup diperoleh. Penjelasan formal tentang syarat ini diberikan pada Bab 16.

Contoh Angka Perbandingan Tiga Arah

Misalkan pada titik tertentu gradiennya $\nabla S = (2, 1)^\top$. Arah gradien negatif adalah $(-2, -1)^\top$. Jika matriks Gauss–Newton memperbesar skala parameter pertama dan mengecilkan skala parameter kedua, arah Gauss–Newton dapat berbentuk $(-1, -0.2)^\top$. Keduanya memenuhi

$$\nabla S^\top(-\nabla S) = -5 < 0, \qquad \nabla S^\top \boldsymbol{p}_{\mathrm{GN}} = -2.2 < 0.$$

Jadi, kedua arah sama-sama menurunkan fungsi pada awal gerakan. Namun, panjang langkah yang aman dapat berbeda. Arah Gauss–Newton biasanya lebih dekat ke lembah fungsi, sedangkan arah gradien negatif lebih konservatif ketika skala parameter sangat berbeda.

Proposisi 1 (Arah Gauss–Newton adalah Arah Turun)**.** *Jika J_k berperingkat penuh kolom, maka arah $\boldsymbol{p}_k$ dari* (3.14) *merupakan arah turun (*descent direction*) untuk $S(\boldsymbol{\theta})$ di $\boldsymbol{\theta}_k$, kecuali $\boldsymbol{\theta}_k$ adalah titik stasioner.*

Bukti. Suatu arah $\boldsymbol{p}$ disebut arah turun jika $\nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p} < 0$, yaitu jika turunan berarah S pada arah $\boldsymbol{p}$ bernilai negatif. Mari kita hitung hasil kali dalam ini, memakai $\nabla S(\boldsymbol{\theta}_k) = J_k^\top \boldsymbol{r}_k$ dari (2.4) dan $\boldsymbol{p}_k$ dari (3.14):

$$\begin{aligned} \nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k &= \left(J_k^\top \boldsymbol{r}_k\right)^\top \left[-\left(J_k^\top J_k\right)^{-1} J_k^\top \boldsymbol{r}_k\right] \\ &= -\boldsymbol{r}_k^\top J_k \left(J_k^\top J_k\right)^{-1} J_k^\top \boldsymbol{r}_k \\ &= -\left\| \left(J_k^\top J_k\right)^{-1/2} J_k^\top \boldsymbol{r}_k \right\|^2 \le 0, \end{aligned} \tag{5.4}$$

dan kesamaan pada (5.4) hanya terjadi jika $J_k^\top \boldsymbol{r}_k = \boldsymbol{0}$, artinya $\boldsymbol{\theta}_k$ adalah titik stasioner (bandingkan (2.5)). Karena $J_k^\top J_k$ definit positif untuk J_k berperingkat penuh kolom, pertidaksamaan ini bersifat tegas selama $\boldsymbol{\theta}_k$ bukan titik stasioner. Jadi, $\boldsymbol{p}_k$ memang selalu arah turun. □

Sifat ini penting secara praktis. Karena $\boldsymbol{p}_k$ selalu merupakan arah turun, kita dapat menjamin $S(\boldsymbol{\theta}_{k+1}) < S(\boldsymbol{\theta}_k)$ dengan memilih α_k yang cukup kecil me-

lalui pencarian garis. Jaminan ini tetap berlaku meskipun aproksimasi Hessian yang digunakan, yaitu $J^\top J$, tidak sama persis dengan H. Namun, penurunan nilai fungsi belum tentu berlangsung cepat. Pertanyaan tentang kecepatannya dibahas pada bagian berikutnya.

5.3 Prosedur Pemeriksaan pada Setiap Iterasi

Kita dapat menerapkan pemeriksaan berikut pada data kecil maupun eksperimen publik. Urutannya sengaja dibuat sederhana agar penyebab kegagalan dapat dilacak satu per satu.

1. Hitung $\boldsymbol{r}_k$, J_k, dan gradien $\boldsymbol{g}_k = J_k^\top \boldsymbol{r}_k$.
2. Selesaikan $J_k^\top J_k \boldsymbol{p}_k = -\boldsymbol{g}_k$ melalui QR atau SVD.
3. Periksa hasil kali $\boldsymbol{g}_k^\top \boldsymbol{p}_k$. Nilai negatif menunjukkan arah turun.
4. Coba $\alpha_k = 1$ dan hitung $S(\boldsymbol{\theta}_k + \alpha_k \boldsymbol{p}_k)$.
5. Jika nilai tersebut tidak turun, kecilkan α_k sampai syarat penurunan dipenuhi.
6. Simpan S_k, $\|\boldsymbol{g}_k\|$, $\|\boldsymbol{p}_k\|$, dan α_k untuk membuat kurva pembelajaran.

Jika langkah selalu diperkecil sampai sangat kecil, masalahnya biasanya bukan pada rumus turunan, melainkan pada skala parameter, kekondisian Jacobian, atau tebakan awal yang terlalu jauh. Tiga kemungkinan itu akan muncul kembali ketika kita membahas SVD pada Bab 4, regularisasi pada Bab 7, dan pencarian garis pada Bab 16.

Bab 6

Seberapa Cepat Metode Ini Konvergen?

6.1 Dua Besaran Kunci: Ketidaklinearan dan Kekondisian

Pada Bab 3, kita melihat bahwa Hessian eksak dapat ditulis sebagai $H = J^\top J + M$. Di sini, $M(\boldsymbol{\theta}) = \sum_i r_i(\boldsymbol{\theta}) \nabla^2 r_i(\boldsymbol{\theta})$ adalah suku yang diabaikan oleh Gauss–Newton. Kecepatan konvergensi tentu dipengaruhi oleh besar kecilnya suku tersebut dibandingkan dengan $J^\top J$. Dengan memakai $\sigma_{\min}$ dari SVD pada Bab 4, kita definisikan dua konstanta pada bola $\mathcal{B}(\boldsymbol{\theta}^*, \rho)$ di sekitar solusi $\boldsymbol{\theta}^*$:

$$\gamma \;:=\; \text{konstanta Lipschitz } J(\cdot) \text{ pada } \mathcal{B}(\boldsymbol{\theta}^*, \rho), \text{ yaitu}$$
$$\|J(\boldsymbol{u}) - J(\boldsymbol{v})\| \le \gamma \|\boldsymbol{u} - \boldsymbol{v}\| \;\forall \boldsymbol{u}, \boldsymbol{v} \in \mathcal{B}(\boldsymbol{\theta}^*, \rho), \tag{6.1}$$

$$\kappa \;:=\; \frac{1}{\sigma} \sup_{\boldsymbol{\theta} \in \mathcal{B}(\boldsymbol{\theta}^*, \rho)} \|M(\boldsymbol{\theta})\|, \qquad \sigma \;:=\; \sigma_{\min}(J(\boldsymbol{\theta}^*)) \,. \tag{6.2}$$

Konstanta κ ini kita sebut **faktor ketidaklinearan-residu**. Nilainya besar jika residu $r_i(\boldsymbol{\theta}^*)$ besar (artinya model tidak cocok sempurna dengan data), atau jika model sangat melengkung ($\nabla^2 r_i$ besar) relatif terhadap σ.

6.2 Membaca Laju Konvergensi dari Data Iterasi

Teorema lokal memberikan gambaran matematis. Dalam eksperimen, kita biasanya hanya melihat tabel iterasi dan kurva pembelajaran. Hubungan keduanya dapat dibuat eksplisit. Jika rasio

$$q_k = \frac{\|\boldsymbol{e}_{k+1}\|}{\|\boldsymbol{e}_k\|} \tag{6.3}$$

menetap di sekitar suatu angka kurang dari satu, perilakunya tampak linear. Jika q_k terus mengecil seiring galat mengecil, perilakunya mendekati kuadratik. Nilai q_k pada (6.3) tidak harus dihitung dari solusi eksak. Pada eksperimen nyata, kita dapat menggantinya dengan rasio norma gradien atau rasio perubahan fungsi.

Hubungan antara gejala, dugaan perilaku, dan tindakan perbaikan dirangkum pada Tabel 6.1.

Tabel 6.1: Petunjuk praktis untuk membaca konvergensi

Gejala pada tabel iterasi	Dugaan perilaku	Tindakan yang disarankan
S_k turun cepat dan $\|\boldsymbol{g}_k\|$ ikut turun	Gauss–Newton berada di wilayah lokal yang baik	Pertahankan langkah dan periksa kriteria penghentian
S_k turun, tetapi rasio perubahan hampir tetap	Konvergensi linear	Periksa residu akhir dan nilai singular terkecil
S_k naik atau berosilasi	Langkah penuh terlalu agresif	Gunakan pencarian garis atau Levenberg–Marquardt
S_k hampir tidak berubah dan $\|\boldsymbol{g}_k\|$ besar	Arah tidak informatif atau sistem buruk	Periksa skala data, peringkat, dan kondisi Jacobian

Tabel 6.1 menunjukkan mengapa satu angka akurasi akhir belum cukup. Dua metode dapat memiliki nilai objektif akhir yang mirip, tetapi salah satunya mencapainya dengan langkah yang stabil, sedangkan yang lain sempat meledak lalu pulih. Kurva pembelajaran dan catatan langkah α_k membantu membedakan kedua keadaan tersebut.

6.3 Teorema Konvergensi Lokal

Teorema 1 (Konvergensi Lokal Gauss–Newton)**.** *Misalkan $\boldsymbol{\theta}^*$ solusi (2.5), J Lipschitz kontinu dengan konstanta γ pada $\mathcal{B}(\boldsymbol{\theta}^*, \rho)$ seperti pada (6.1), dan $J(\boldsymbol{\theta}^*)$ berperingkat penuh kolom dengan $\sigma_{\min}(J(\boldsymbol{\theta}^*)) = \sigma > 0$. Jika κ pada (6.2) memenuhi $\kappa < 1$, maka terdapat $\bar{\rho} \in (0, \rho]$ sedemikian hingga untuk setiap $\boldsymbol{\theta}_0 \in \mathcal{B}(\boldsymbol{\theta}^*, \bar{\rho})$, barisan Gauss–Newton murni ($\alpha_k = 1$) konvergen ke $\boldsymbol{\theta}^*$ dengan laju*

$$\|\boldsymbol{e}_{k+1}\| \leq \left[\kappa + \frac{\gamma\,\|J(\boldsymbol{\theta}^*)\|}{2\sigma(1-\kappa)}\|\boldsymbol{e}_k\|\right]\|\boldsymbol{e}_k\|, \qquad \boldsymbol{e}_k := \boldsymbol{\theta}_k - \boldsymbol{\theta}^*. \tag{6.4}$$

*Jika $\kappa = 0$ (kasus residu nol, $\boldsymbol{r}(\boldsymbol{\theta}^*) = \mathbf{0}$), konvergensi bersifat **kuadratik**; jika $0 < \kappa < 1$, konvergensi bersifat **linear** dengan rasio asimtotik κ.*

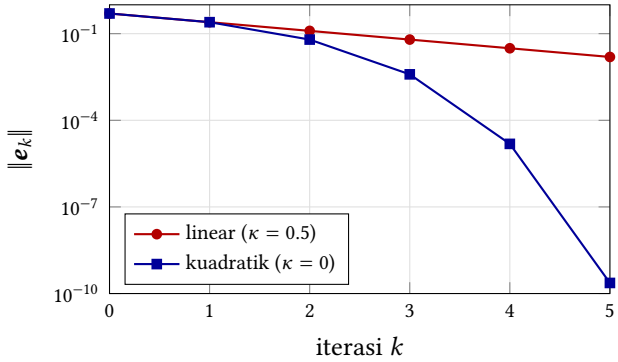
▲ Pendalaman Teknis

Bukti Teorema 1 berikut cukup panjang (lima langkah, memakai ekspansi Taylor terintegralkan dan estimasi Lipschitz) dan tidak dibutuhkan untuk memahami bagian-bagian selanjutnya. Analisis sejenis dapat ditemukan pada Dennis dan Schnabel (1996) dan Nocedal dan Wright (2006). Bukti lengkapnya dipindahkan ke **Lampiran A** agar alur baca utama tetap ringan. Kita dapat langsung melompat ke Catatan setelah pernyataan Teorema, yang merangkum maknanya dalam bahasa biasa.

Sketsa bukti. Bukti lengkap di Lampiran A. Ide utamanya: tulis $\boldsymbol{e}_k = \boldsymbol{\theta}_k - \boldsymbol{\theta}^*$, lalu ekspansi Taylor residu $\boldsymbol{r}_k$ di sekitar $\boldsymbol{\theta}^*$ dipecah menjadi suku yang berkaitan dengan $M(\boldsymbol{\theta}^*)$ (mengecil sebanding κ) dan suku sisa yang mengecil sebanding kuadrat $\|\boldsymbol{e}_k\|$ (berkat kekontinuan Lipschitz J). Menggabungkan keduanya dengan kontrol atas $(J_k^\top J_k)^{-1}$ menghasilkan bentuk (6.4). Detail lengkap lima langkahnya ada di Lampiran A. □

Catatan 1. *Teorema 1 menjelaskan secara kuantitatif mengapa Gauss–Newton bekerja sangat baik ketika $\kappa \approx 0$. Konvergensi melambat saat κ mendekati 1 dan dapat gagal ketika syarat cukup $\kappa < 1$ tidak terpenuhi. Situasi tersebut memotivasi regularisasi pada Bab 7, terutama ketika $J^\top J$ mendekati singular.*

Selisih antara laju linear dan kuadratik tampak jelas pada grafik. Gambar 6.1 membandingkan dua barisan galat yang sama-sama dimulai dari $\|\boldsymbol{e}_0\| = 0.5$. Barisan pertama mengikuti $\|\boldsymbol{e}_{k+1}\| = \kappa\|\boldsymbol{e}_k\|$ dengan $\kappa = 0.5$, sedangkan barisan kedua mengikuti $\|\boldsymbol{e}_{k+1}\| = \|\boldsymbol{e}_k\|^2$ dengan $\kappa = 0$.



Gambar 6.1: Perbandingan penurunan galat pada konvergensi linear dan kuadratik

Pada sumbu-y logaritmik, barisan linear tampak sebagai garis lurus yang menurun perlahan. Barisan kuadratik turun jauh lebih tajam. Pada iterasi kelima, galatnya sudah sekitar 2.3×10^{-10}. Satu iterasi kuadratik lagi akan membawanya ke sekitar 5.4×10^{-20}, lebih kecil daripada ketelitian aritmetika presisi ganda. Perbedaan ini menjelaskan mengapa kasus residu nol sangat istimewa. Setelah iterasi cukup dekat dengan solusi, jumlah digit yang akurat dapat bertambah hampir dua kali lipat pada setiap langkah.

6.4 Apa yang Perlu Dilaporkan dalam Eksperimen

Agar klaim tentang konvergensi dapat diperiksa, laporan eksperimen sebaiknya memuat empat hal, yaitu (1) tebakan awal dan kriteria berhenti; (2) nilai fungsi dan norma gradien pada setiap iterasi atau pada beberapa iterasi yang representatif; (3) metode penyelesaian sistem linear yang dipakai; serta (4) keterangan apakah langkah penuh diterima atau diperpendek.

Dengan empat informasi tersebut, kita dapat membedakan tiga keadaan.

Metode mungkin cepat karena masalahnya memang memiliki residu hampir nol. Metode mungkin tampak lambat karena data tidak cocok sempurna dengan model. Metode juga mungkin terlihat tidak stabil karena sistem normal dibentuk secara eksplisit. Bab berikutnya membahas cara mengubah kasus terakhir menjadi eksperimen yang lebih aman.

Bab 7

Mengatasi Ketidakstabilan: Regularisasi Levenberg–Marquardt

Analogi

Mari kita kembali ke analogi bukit berkabut pada Subbab 3.1. Metode Newton melompat ke dasar mangkuk perkiraan dengan asumsi bahwa bentuk mangkuk tersebut dapat dipercaya. Namun, keadaan dapat berbeda jika kita berada di punggungan sempit atau di daerah yang hampir datar pada salah satu arah. Perkiraan mangkuk dapat meleset dan langkah Newton justru menjauh dari lembah. Levenberg–Marquardt mengambil sikap yang lebih hati-hati. Metode ini membatasi panjang langkah melalui radius kepercayaan Δ_k. Di daerah yang aman, langkah dapat dibuat panjang. Di daerah yang mencurigakan, langkah dipendekkan dan diarahkan ke arah penurunan tercuram yang lebih aman, meskipun lebih lambat.

Dua bab sebelumnya memperlihatkan dua sisi dari masalah yang sama. Secara numerik, $J_k^\top J_k$ sulit dihitung dengan aman ketika $\sigma_{\min}(J_k)$ kecil. Secara analitis, konvergensi juga melambat atau gagal ketika $\sigma_{\min}(J(\boldsymbol{\theta}^*))$ kecil sehingga κ membesar. Akar masalahnya sama, yaitu $J_k^\top J_k$ mendekati singular. Metode Levenberg–Marquardt mengatasinya dengan menambahkan suku redaman (*damping*) $\lambda_k \geq 0$ pada persamaan normal (Levenberg, 1944; Marquardt,

1963; Moré, 1978; Nielsen, 1999):

$$\left(J_k^\top J_k + \lambda_k I\right) \boldsymbol{p}_k = -J_k^\top \boldsymbol{r}_k. \tag{7.1}$$

Contoh Angka

Misalkan pada satu iterasi diperoleh

$$J_k^\top J_k = \begin{pmatrix} 0.01 & 0 \\ 0 & 4 \end{pmatrix}, \qquad J_k^\top \boldsymbol{r}_k = \begin{pmatrix} 1 \\ 2 \end{pmatrix}.$$

Tanpa redaman, langkah Gauss–Newton memenuhi $\boldsymbol{p}_{\mathrm{GN}} = (-100, -0.5)^\top$. Komponen pertama sangat besar karena entri 0.01 membuat matriks tersebut hampir singular. Jika dipilih $\lambda_k = 1$, sistemnya menjadi

$$\begin{pmatrix} 1.01 & 0 \\ 0 & 5 \end{pmatrix} \boldsymbol{p} = -\begin{pmatrix} 1 \\ 2 \end{pmatrix},$$

sehingga $\boldsymbol{p} \approx (-0.99, -0.4)^\top$. Redaman tidak mengubah tujuan optimisasi. Redaman hanya mencegah langkah yang terlalu berani ketika informasi kelengkungan tidak dapat dipercaya.

Kode MATLAB: Langkah Levenberg–Marquardt

```
clear; clc;
x = (0:3)';
y = [2.1; 3.9; 7.8; 15.2];
theta = [2.0; 0.6];
lambda = 1e-2;
e = exp(theta(2)*x);
r = y - theta(1)*e;
J = [-e, -theta(1)*x.*e];
n = size(J,2);
p = -(J'*J + lambda*eye(n)) \ (J'*r);
disp(p)
```

Menambahkan $\lambda_k I$ ini menaikkan seluruh nilai eigen $J_k^\top J_k$ (yaitu σ_i^2, dari SVD

di Bab 4) sebesar λ_k, sehingga matriks $\left(J_k^\top J_k + \lambda_k I\right)$ terjamin selalu invertibel, bahkan ketika J_k kehilangan peringkat penuh.

Persamaan (7.1) bukan sekadar modifikasi *ad hoc*. Persamaan tersebut muncul secara alami dari masalah optimisasi berkendala. Kita tinjau kembali subproblem Gauss–Newton (3.10), kali ini dengan kendala daerah kepercayaan (*trust region*). Aproksimasi linear (3.9) hanya dipercaya selama langkah $\boldsymbol{p}$ tidak terlalu jauh dari titik saat ini. Jarak tersebut dibatasi oleh radius Δ_k:

$$\boldsymbol{p}_k = \arg\min_{\boldsymbol{p}} \frac{1}{2}\left\|\boldsymbol{r}_k + J_k\boldsymbol{p}\right\|_2^2 \quad \text{terhadap kendala} \quad \|\boldsymbol{p}\|_2 \le \Delta_k. \tag{7.2}$$

Proposisi 2 (Kondisi KKT untuk Subproblem Daerah Kepercayaan)**.** *Solusi* $\boldsymbol{p}_k$ *dari* (7.2) *memenuhi, untuk suatu* $\lambda_k \ge 0$*:*

$$\left(J_k^\top J_k + \lambda_k I\right)\boldsymbol{p}_k = -J_k^\top \boldsymbol{r}_k, \qquad \lambda_k(\|\boldsymbol{p}_k\| - \Delta_k) = 0, \qquad \|\boldsymbol{p}_k\| \le \Delta_k. \tag{7.3}$$

Bukti. Fungsi Lagrange untuk (7.2) adalah

$$\mathcal{L}(\boldsymbol{p}, \lambda) = \frac{1}{2}\left\|\boldsymbol{r}_k + J_k\boldsymbol{p}\right\|_2^2 + \frac{\lambda}{2}\left(\|\boldsymbol{p}\|_2^2 - \Delta_k^2\right). \tag{7.4}$$

Syarat stasioner $\nabla_{\boldsymbol{p}}\mathcal{L} = \boldsymbol{0}$ untuk (7.4) memberi

$$J_k^\top(\boldsymbol{r}_k + J_k\boldsymbol{p}) + \lambda\boldsymbol{p} = \boldsymbol{0} \iff \left(J_k^\top J_k + \lambda I\right)\boldsymbol{p} = -J_k^\top \boldsymbol{r}_k, \tag{7.5}$$

yaitu (7.5), sementara syarat komplementaritas kendur $\lambda(\|\boldsymbol{p}\| - \Delta_k) = 0$ mengikuti langsung dari teori KKT untuk masalah konveks berkendala pertidaksamaan (di sini fungsi objektifnya kuadrat konveks, dan daerah layaknya berupa bola konveks). □

Jadi, λ_k berperan sebagai pengganda Lagrange yang berkaitan langsung dengan radius kepercayaan Δ_k. Semakin kecil Δ_k, semakin besar λ_k yang diperlukan. Saat $\lambda_k \to 0$, (7.3) kembali menjadi persamaan normal Gauss–Newton. Sebaliknya, saat $\lambda_k \to \infty$, arah langkah mendekati arah penurunan tercuram yang diskalakan, yaitu $\boldsymbol{p}_k \approx -\frac{1}{\lambda_k} J_k^\top \boldsymbol{r}_k$. Dengan demikian, Levenberg–Marquardt

dapat dipandang sebagai penghubung adaptif antara Gauss–Newton dan penurunan gradien.

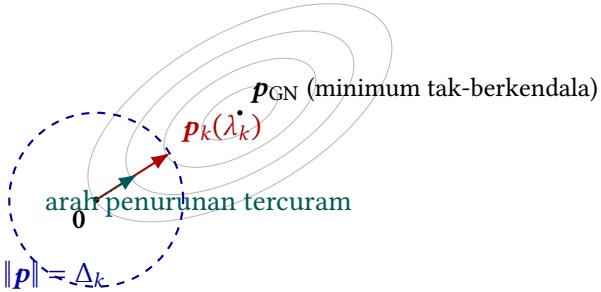
Lemma 1 (Monotonisitas Norma Solusi terhadap λ)**.** *Untuk $\lambda \geq 0$, fungsi $\lambda \mapsto \|\boldsymbol{p}(\lambda)\|$ dengan $\boldsymbol{p}(\lambda) = -\left(J_k^\top J_k + \lambda I\right)^{-1} J_k^\top \boldsymbol{r}_k$ bersifat monoton tak-naik. Dengan memakai SVD $J_k = U\Sigma V^\top$ dari Bab 4, diperoleh*

$$\|\boldsymbol{p}(\lambda)\|^2 = \sum_{i=1}^{n} \left(\frac{\sigma_i \left[U^\top \boldsymbol{r}_k\right]_i}{\sigma_i^2 + \lambda} \right)^2, \tag{7.6}$$

yang jelas menurun terhadap λ pada tiap suku penjumlahannya.

Sifat monoton inilah yang mendasari algoritma pencarian λ_k (misalnya algoritma Moré), yang menyesuaikan λ_k secara iteratif agar $\|\boldsymbol{p}_k\| \approx \Delta_k$ ketika kendala sedang aktif.

Gambar 7.1 memperlihatkan ruang $\boldsymbol{p}$ di sekitar $\boldsymbol{\theta}_k$. Kontur elips adalah garis-garis ketinggian dari model kuadratik lokal $\phi(\boldsymbol{p})$ pada (3.10). Bentuknya elips karena $J_k^\top J_k$ umumnya tidak isotropik. Jika daerah kepercayaan Δ_k terlalu kecil untuk memuat minimum tak-berkendala $\boldsymbol{p}_{\mathrm{GN}}$, solusi berhenti di batas lingkaran dan membelok ke arah penurunan tercuram.



Gambar 7.1: Hubungan antara langkah Gauss–Newton, daerah kepercayaan, dan arah penurunan tercuram

Terlihat bahwa $\boldsymbol{p}_k(\lambda_k)$ (panah merah) berada tepat di batas lingkaran radius Δ_k. Letaknya berada di antara arah Gauss–Newton murni dan arah penurunan

tercuram, sesuai kesimpulan setelah Proposisi 2.

7.1 Strategi Praktis Menyesuaikan λ_k

Pembahasan Levenberg–Marquardt di atas, yang juga digunakan pada Bab 13–14, sejauh ini hanya menyebut aturan penyesuaian λ_k secara umum. Nilai tersebut diperkecil jika langkah berhasil dan diperbesar jika langkah gagal. Gambaran ini masih menyisakan pertanyaan praktis. Seberapa besar perubahan tersebut harus dilakukan? Dua jawaban klasik digunakan secara luas dalam implementasi Levenberg–Marquardt. Jawaban pertama sangat sederhana, sedangkan jawaban kedua lebih halus.

Strategi Marquardt (1963): Faktor Kelipatan Tetap. Versi paling awal (Marquardt, 1963) dan paling sederhana hanya memakai satu faktor tetap $\nu > 1$ (biasanya $\nu = 10$):

Strategi Marquardt

1. Hitung $\boldsymbol{p}_k$ dari (7.1) memakai λ_k saat ini.
2. Jika $S(\boldsymbol{\theta}_k + \boldsymbol{p}_k) < S(\boldsymbol{\theta}_k)$: terima langkah, $\lambda_{k+1} = \lambda_k / \nu$, lanjut ke iterasi berikutnya.
3. Jika tidak: tolak langkah (jangan perbarui $\boldsymbol{\theta}_k$), $\lambda_k \leftarrow \nu\,\lambda_k$, kembali ke langkah 1.

Kode MATLAB: Strategi Marquardt

```
clear; clc;
x = (0:3)';
y = [2.1; 3.9; 7.8; 15.2];
theta = [2.0; 0.6];
lambda = 1e-2;
f = @(u,t) t(1)*exp(t(2)*u);
S = @(t) 0.5*sum((y-f(x,t)).^2);
e = exp(theta(2)*x);
```

Kode MATLAB: Strategi Marquardt *(lanjutan)*

```
r = y - theta(1)*e;
J = [-e, -theta(1)*x.*e];
n = length(theta);
nu = 10;
while true
    p = -(J'*J + lambda*eye(n)) \ (J'*r);
    if S(theta + p) < S(theta)
        theta = theta + p;
        lambda = lambda / nu;
        break;
    else
        lambda = lambda * nu;
    end
end
disp(theta)
disp(lambda)
```

Kelemahan strategi ini adalah kekakuannya: faktor ν yang sama dipakai tidak peduli apakah langkah "hampir berhasil" atau "gagal total". Keduanya diperlakukan sama persis. Strategi kedua memperbaiki ini dengan mengukur *seberapa baik* suatu langkah, bukan sekadar berhasil-atau-tidak.

Strategi Nielsen (1999): Rasio Keuntungan (*Gain Ratio*). Gagasannya: bandingkan penurunan S yang *sungguhan terjadi* dengan penurunan yang *diprediksi* oleh model kuadratik lokal $\phi(\boldsymbol{p})$ pada (3.10). **Rasio keuntungan** didefinisikan sebagai

$$\rho_k := \frac{S(\boldsymbol{\theta}_k) - S(\boldsymbol{\theta}_k + \boldsymbol{p}_k)}{L(\mathbf{0}) - L(\boldsymbol{p}_k)}, \tag{7.7}$$

dengan $L(\boldsymbol{p}) := S(\boldsymbol{\theta}_k) + \boldsymbol{p}^\top \boldsymbol{g}_k + \frac{1}{2}\boldsymbol{p}^\top (J_k^\top J_k)\boldsymbol{p}$ sebagai model kuadratik Gauss–Newton biasa (tanpa suku redaman λ) dan $\boldsymbol{g}_k := J_k^\top \boldsymbol{r}_k$. Penyebut (7.7) dapat disederhanakan menjadi bentuk yang mudah dihitung: karena $\boldsymbol{p}_k$ memenuhi persamaan LM $(J_k^\top J_k + \lambda_k I)\boldsymbol{p}_k = -\boldsymbol{g}_k$, kita memperoleh $J_k^\top J_k\, \boldsymbol{p}_k = -\boldsymbol{g}_k - \lambda_k \boldsymbol{p}_k$, sehingga

$$L(\mathbf{0}) - L(\boldsymbol{p}_k) = -\boldsymbol{p}_k^\top \boldsymbol{g}_k - \tfrac{1}{2}\boldsymbol{p}_k^\top (J_k^\top J_k)\boldsymbol{p}_k$$

$$= -\boldsymbol{p}_k^\top \boldsymbol{g}_k - \tfrac{1}{2}\boldsymbol{p}_k^\top(-\boldsymbol{g}_k - \lambda_k \boldsymbol{p}_k)$$
$$= -\tfrac{1}{2}\boldsymbol{p}_k^\top \boldsymbol{g}_k + \tfrac{1}{2}\lambda_k \boldsymbol{p}_k^\top \boldsymbol{p}_k = \tfrac{1}{2}\boldsymbol{p}_k^\top(\lambda_k \boldsymbol{p}_k - \boldsymbol{g}_k). \tag{7.8}$$

Kode MATLAB: Menghitung Rasio Keuntungan ρ_k

```
clear; clc;
x = (0:3)';
y = [2.1; 3.9; 7.8; 15.2];
theta = [2.0; 0.6];
lambda = 1e-2;
f = @(u,t) t(1)*exp(t(2)*u);
S = @(t) 0.5*sum((y-f(x,t)).^2);
e = exp(theta(2)*x);
r = y - theta(1)*e;
J = [-e, -theta(1)*x.*e];
g = J'*r;
p = -(J'*J + lambda*eye(2)) \ g;
pred_decrease = 0.5 * p' * (lambda*p - g);
actual_decrease = S(theta) - S(theta + p);
rho = actual_decrease / pred_decrease;
disp(rho)
```

Rasio ρ_k yang dekat dengan 1 berarti model kuadratik memprediksi penurunan dengan akurat. Dalam keadaan itu, λ_k dapat dikecilkan lebih agresif. Sebaliknya, rasio yang kecil atau negatif menunjukkan bahwa model lokal meleset sehingga λ_k perlu dibesarkan. Nielsen mengusulkan aturan berikut. Faktor pembesar ν ikut meningkat setiap kali langkah gagal berturut-turut.

Strategi Nielsen

1. Inisialisasi $\nu = 2$. Hitung $\boldsymbol{p}_k$ dari (7.1), lalu ρ_k dari (7.7).
2. Jika $\rho_k > 0$ (langkah membaik): terima $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{p}_k$, perbarui
$$\lambda_{k+1} = \lambda_k \cdot \max\!\left(\tfrac{1}{3},\, 1 - (2\rho_k - 1)^3\right), \qquad \nu \leftarrow 2.$$
3. Jika $\rho_k \le 0$ (langkah memburuk): tolak langkah, $\lambda_k \leftarrow \nu\lambda_k$, $\nu \leftarrow 2\nu$, kembali ke langkah 1 dengan $\boldsymbol{\theta}_k$ yang sama.

Kode MATLAB: Strategi Nielsen

```
clear; clc;
x = (0:3)';
y = [2.1; 3.9; 7.8; 15.2];
theta = [2.0; 0.6];
lambda = 1e-2;
f = @(u,t) t(1)*exp(t(2)*u);
S = @(t) 0.5*sum((y-f(x,t)).^2);
e = exp(theta(2)*x);
r = y - theta(1)*e;
J = [-e, -theta(1)*x.*e];
n = length(theta);
nu = 2;
while true
    p = -(J'*J + lambda*eye(n)) \ (J'*r);
    pred = 0.5 * p'*(lambda*p - J'*r);
    rho  = (S(theta) - S(theta+p)) / pred;
    if rho > 0
        theta = theta + p;
        lambda = lambda * max(1/3, 1-(2*rho-1)^3);
        nu = 2;
        break;
    else
        lambda = lambda * nu;
        nu = 2 * nu;
    end
end
disp(theta)
disp(lambda)
```

Faktor $\max(1/3,\ 1-(2\rho_k-1)^3)$ pada langkah 2 mengatur perubahan redaman berdasarkan mutu model lokal. Ketika $\rho_k \approx 1$, prediksi model sangat akurat dan λ_k dapat dikecilkan hingga sepertiganya. Ketika ρ_k hanya sedikit positif, model masih meleset meskipun nilai objektif menurun, sehingga λ_k hampir tidak diubah. Berbeda dari strategi Marquardt yang memakai faktor tetap, strategi Nielsen menyesuaikan λ_k secara halus. Pada masalah dengan kelengkungan yang berubah cepat, penyesuaian ini biasanya mengurangi jumlah percobaan langkah yang ditolak.

7.2 Konvergensi Global Levenberg–Marquardt

▲ Pendalaman Teknis

Subbab ini melengkapi Teorema 1 untuk Gauss–Newton murni dan Teorema 3 untuk pencarian garis. Hasilnya harus dibaca bersama asumsi teorema. Intinya, strategi redaman mencegah penolakan langkah berlangsung tanpa akhir. Di bawah syarat keterbatasan dan kemulusan yang dinyatakan kemudian, terdapat barisan bagian dengan norma gradien menuju nol.

Baik Teorema 1 maupun Teorema 3 tidak langsung berlaku untuk algoritma LM: yang pertama mengasumsikan $\alpha_k = 1$ tanpa redaman, yang kedua mengasumsikan mekanisme pencarian garis (menyesuaikan panjang langkah α_k pada arah tetap), sementara LM justru menyesuaikan *arah* langkah melalui λ_k. Untungnya, klaim serupa tetap dapat dibuktikan, dimulai dari fakta bahwa langkah LM *selalu dapat diterima* asalkan λ_k dibesarkan cukup jauh.

Proposisi 3 (Keterjaminan Penerimaan Langkah)**.** *Jika $\nabla S(\boldsymbol{\theta}_k) \neq \mathbf{0}$, maka terdapat $\bar{\lambda}$ sedemikian hingga untuk semua $\lambda \geq \bar{\lambda}$, langkah $\boldsymbol{p}(\lambda) = -(J_k^\top J_k + \lambda I)^{-1}\nabla S(\boldsymbol{\theta}_k)$ memenuhi $S(\boldsymbol{\theta}_k + \boldsymbol{p}(\lambda)) < S(\boldsymbol{\theta}_k)$.*

Bukti. Tulis $\boldsymbol{g} := \nabla S(\boldsymbol{\theta}_k)$. Untuk λ besar, $(J_k^\top J_k + \lambda I)^{-1} = \frac{1}{\lambda}\big(I + \frac{1}{\lambda}J_k^\top J_k\big)^{-1} = \frac{1}{\lambda}I - \frac{1}{\lambda^2}J_k^\top J_k + O(\lambda^{-3})$ (deret Neumann), sehingga

$$\boldsymbol{p}(\lambda) = -\tfrac{1}{\lambda}\boldsymbol{g} + O(\lambda^{-2}).$$

Karena S terdiferensialkan kontinu, ekspansi Taylor orde satu memberi

$$\begin{aligned} S(\boldsymbol{\theta}_k + \boldsymbol{p}(\lambda)) &= S(\boldsymbol{\theta}_k) + \boldsymbol{g}^\top \boldsymbol{p}(\lambda) + o(\|\boldsymbol{p}(\lambda)\|) \\ &= S(\boldsymbol{\theta}_k) - \tfrac{1}{\lambda}\|\boldsymbol{g}\|^2 + o(\lambda^{-1}), \end{aligned}$$

memakai $\boldsymbol{g}^\top \boldsymbol{p}(\lambda) = -\frac{1}{\lambda}\|\boldsymbol{g}\|^2 + O(\lambda^{-2})$ dan $\|\boldsymbol{p}(\lambda)\| = O(\lambda^{-1})$. Karena suku $o(\lambda^{-1})$ mengecil lebih cepat daripada λ^{-1}, untuk λ yang cukup besar, nilai mutlak suku

ini kurang dari $\frac{1}{2\lambda}\|\boldsymbol{g}\|^2$, sehingga

$$S(\boldsymbol{\theta}_k + \boldsymbol{p}(\lambda)) \le S(\boldsymbol{\theta}_k) - \tfrac{1}{2\lambda}\|\boldsymbol{g}\|^2 < S(\boldsymbol{\theta}_k). \qquad \square$$

Proposisi ini penting secara praktis. Proposisi tersebut menjamin bahwa strategi Marquardt maupun strategi Nielsen tidak akan menolak langkah selamanya. Marquardt memperbesar λ_k berkali-kali lipat setiap kali gagal, sedangkan Nielsen memperbesarnya secara semakin agresif melalui $\nu \leftarrow 2\nu$. Cepat atau lambat, λ_k akan cukup besar sehingga langkah dapat diterima. Hasil ini menjadi dasar teorema konvergensi global berikut.

Teorema 2 (Konvergensi Global Levenberg–Marquardt)**.** *Misalkan S terbatas di bawah dan ∇S Lipschitz kontinu pada himpunan level $\{\boldsymbol{\theta} : S(\boldsymbol{\theta}) \le S(\boldsymbol{\theta}_0)\}$. Dengan asumsi tersebut, barisan $\{\boldsymbol{\theta}_k\}$ yang dibangun oleh algoritma Levenberg–Marquardt (memakai strategi Marquardt atau Nielsen pada Subbab 7.1, dengan $J_k^\top J_k$ terbatas seragam) memenuhi*

$$\liminf_{k\to\infty} \|\nabla S(\boldsymbol{\theta}_k)\| = 0.$$

Bukti lengkap Teorema 2 terdapat di Lampiran C. Polanya serupa dengan Teorema 3: jumlah penurunan S dikontrol oleh fakta bahwa S terbatas di bawah. Perbedaannya, penurunan setiap langkah di sini dijamin oleh Proposisi 3, bukan oleh syarat Wolfe.

Catatan 2. *Teorema 2 tidak menyatakan bahwa seluruh iterasi pasti menuju satu titik minimum tertentu. Kesimpulannya lebih terbatas, yaitu* $\liminf \|\nabla S(\boldsymbol{\theta}_k)\| = 0$ *di bawah asumsi yang disebutkan. Jika iterasi kemudian memasuki lingkungan solusi lokal $\boldsymbol{\theta}^*$ dan $\lambda_k \to 0$, perilakunya mendekati Gauss–Newton murni. Teorema 1 lalu menjelaskan lajunya. Konvergensi bersifat kuadratik ketika $\kappa = 0$ dan linear ketika $\kappa > 0$.*

7.3 Eksperimen: Marquardt versus Nielsen pada MLP

Teorema di atas memberi jaminan asimtotik di bawah asumsi tertentu, tetapi tidak menentukan strategi mana yang lebih hemat. Untuk menjawab pertanya-

an praktis itu, kedua strategi diuji pada MLP dua unit beraktivasi tanh yang mencocokkan $y = \sin(x)$. Parameter awal dibuat sama dengan contoh pada Subbab 13.5, tetapi data yang dipakai adalah delapan titik berjarak sama pada $[-3, 3]$. Pada data ini, $S(\boldsymbol{\theta}_0) \approx 1.380$ dan langkah Gauss–Newton murni menaikkannya menjadi sekitar 725.2. Lonjakan ini serupa dengan yang terjadi pada contoh Subbab 13.5. Marquardt dan Nielsen masing-masing dijalankan sampai memperoleh 40 langkah yang diterima, dengan $\lambda_0 = 10^{-2}$ (Nielsen, 1999).

Langkah Eksperimen

Ikuti tahapan berikut sebelum membaca hasil.

1. Bentuk delapan titik x berjarak sama pada $[-3, 3]$ dan tetapkan $y_i = \sin(x_i)$. Gunakan parameter awal MLP satu lapisan tersembunyi pada Subbab 13.5.
2. Lakukan propagasi maju. Hitung residu dan Jacobian dengan (13.3)–(13.5).
3. Bentuk $\boldsymbol{g} = J^{\top}\boldsymbol{r}$, kemudian selesaikan persamaan LM (7.1).
4. Untuk Marquardt, pilih $\lambda_0 = 10^{-2}$ dan $\nu = 10$. Jika S turun, terima langkah dan bagi λ dengan 10. Jika tidak, tolak langkah dan kalikan λ dengan 10.
5. Untuk Nielsen, pilih $\nu = 2$. Hitung ρ melalui (7.7), terima langkah jika $\rho > 0$, lalu perbarui λ dan ν menurut Subbab 7.1.
6. Hentikan setelah 40 langkah diterima. Simpan S sesudah setiap langkah dan catat jumlah percobaan yang ditolak.

Ringkasan perbandingan kedua strategi disajikan pada Tabel 7.1.

Kedua strategi menghindari lonjakan yang dialami Gauss–Newton murni. Pada langkah pertama, S turun ke sekitar 0.53. Setelah 40 langkah diterima, keduanya berada pada lingkungan nilai objektif yang sebanding, sekitar 2×10^{-5}. Kesamaan ini merupakan hasil eksperimen, bukan jaminan bahwa keduanya

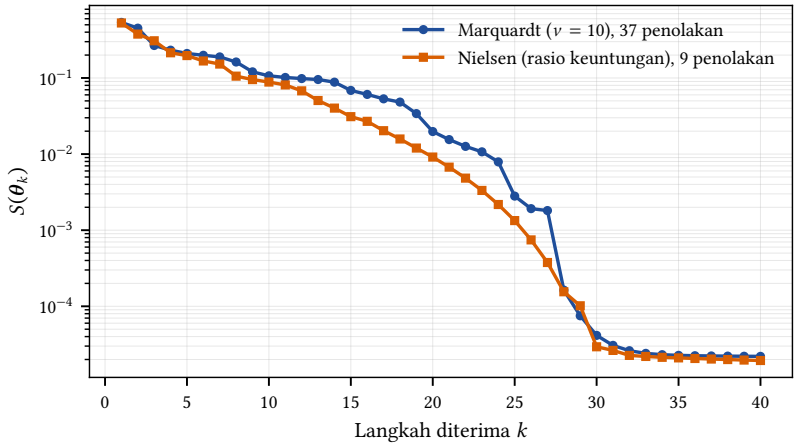
Tabel 7.1: Perbandingan strategi Marquardt dan Nielsen

Strategi	Iterasi berhasil	Total percobaan ditolak	S akhir
Marquardt ($\nu = 10$ tetap)	40	**37**	2.2×10^{-5}
Nielsen (rasio keuntungan)	40	**9**	1.9×10^{-5}

selalu menuju titik stasioner yang sama.

Perbedaan utama terlihat pada efisiensi. Marquardt menolak 37 percobaan, sedangkan Nielsen hanya menolak 9. Dalam eksperimen ini, Nielsen menggunakan sekitar seperempat jumlah penolakan Marquardt. Rasio keuntungan ρ_k membuat perubahan λ_k mengikuti mutu prediksi model lokal, bukan hanya keputusan berhasil atau gagal.

Perbandingan visual lintasan kedua strategi ditunjukkan pada Gambar 7.2.



Gambar 7.2: Perbandingan strategi Marquardt dan Nielsen pada MLP

Catatan 3. *Eksperimen ini dijalankan dengan GNU Octave melalui skrip* `lm_strategy_comparison.m`*. Grafiknya dibuat ulang dengan Python (NumPy dan Matplotlib) memakai algoritma yang sama dan menghasilkan angka yang identik dengan Tabel 7.1. Implementasinya mengikuti algoritma Marquardt dan Nielsen pada Subbab 7.1, termasuk (7.7)–(7.8). Setiap percobaan λ yang ditolak me-*

merlukan satu penyelesaian sistem linear tambahan. Karena itu, jumlah penolakan yang lebih sedikit dapat mengurangi biaya komputasi total.

7.4 Alternatif: Dogleg dan Persamaan Sekuler Moré

Baik strategi Marquardt maupun Nielsen menyesuaikan λ_k secara tidak langsung. Kita memilih λ_k, menghitung langkah, lalu memeriksa mutu hasilnya. Radius kepercayaan Δ_k tidak dikontrol secara eksplisit, meskipun λ_k dan Δ_k saling berkaitan seperti diperlihatkan pada Gambar 7.1. Dua pendekatan klasik berikut menyasar Δ_k secara langsung. Untuk radius yang telah ditentukan, keduanya mencari langkah $\boldsymbol{p}_k$ dengan $\|\boldsymbol{p}_k\| \approx \Delta_k$ atau tepat sama dengannya.

Metode Dogleg: Pendekatan Murah. Ide Powell (1970) ini menghindari penyelesaian sistem linear berulang-ulang dengan membangun lintasan perkiraan yang mudah dihitung dengan memakai dua titik yang sudah kita kenal:

- **Titik Cauchy $\boldsymbol{p}^C$**: minimum model kuadratik $\phi(\boldsymbol{p})$ pada (3.10) sepanjang arah penurunan tercuram $-\boldsymbol{g}_k$ saja (bukan ke segala arah).

Untuk memperoleh rumus titik Cauchy tanpa melompati perhitungannya, tulis $\boldsymbol{g}_k := J_k^\top \boldsymbol{r}_k$ dan batasi langkah pada bentuk $\boldsymbol{p} = -t\boldsymbol{g}_k$, dengan $t \geq 0$. Substitusi ke bentuk kuadratik ϕ menghasilkan fungsi satu peubah

$$q(t) := \phi(-t\boldsymbol{g}_k) = \frac{1}{2}\boldsymbol{r}_k^\top \boldsymbol{r}_k - t\boldsymbol{g}_k^\top \boldsymbol{g}_k + \frac{1}{2}t^2 \boldsymbol{g}_k^\top (J_k^\top J_k)\boldsymbol{g}_k. \tag{7.9}$$

Suku pertama pada (7.9) tidak bergantung pada t. Dengan aturan $d(t^2)/dt = 2t$, turunannya adalah

$$q'(t) = -\boldsymbol{g}_k^\top \boldsymbol{g}_k + t\, \boldsymbol{g}_k^\top (J_k^\top J_k)\boldsymbol{g}_k.$$

Syarat $q'(t) = 0$ memberikan

$$t^* = \frac{\boldsymbol{g}_k^\top \boldsymbol{g}_k}{\boldsymbol{g}_k^\top (J_k^\top J_k)\boldsymbol{g}_k}.$$

Karena $\boldsymbol{p}^C = -t^* \boldsymbol{g}_k$, diperoleh

$$\boldsymbol{p}^C = -\frac{\boldsymbol{g}_k^\top \boldsymbol{g}_k}{\boldsymbol{g}_k^\top (J_k^\top J_k) \boldsymbol{g}_k} \boldsymbol{g}_k. \tag{7.10}$$

- **Titik Gauss–Newton** $\boldsymbol{p}^{GN} = -(J_k^\top J_k)^{-1} \boldsymbol{g}_k$: langkah Gauss–Newton murni dari (3.14).

Kode MATLAB: Titik Cauchy dan Titik Gauss–Newton

```
clear; clc;
J = [1, 0; 0, 2; 1, 1];
r = [1; -1; 0.5];
g = J'*r;
pC  = -(g'*g)/(g'*(J'*J)*g) * g;
pGN = -(J'*J) \ g;
disp([pC, pGN])
```

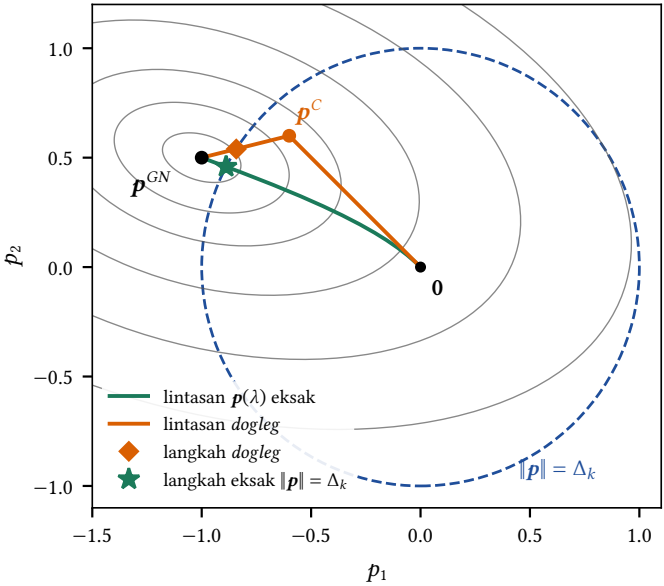
Lintasan *dogleg* adalah garis patah-patah dari $\mathbf{0}$ ke $\boldsymbol{p}^C$, lalu dari $\boldsymbol{p}^C$ ke $\boldsymbol{p}^{GN}$. Lintasan ini mendekati bentuk lengkung lintasan $\boldsymbol{p}(\lambda)$ yang sesungguhnya (lihat Lemma 1) tanpa perlu menghitung $\boldsymbol{p}(\lambda)$ untuk banyak nilai λ. Langkah-langkahnya adalah sebagai berikut.

Algoritma Dogleg

1. Jika $\|\boldsymbol{p}^{GN}\| \le \Delta_k$: ambil $\boldsymbol{p}_k = \boldsymbol{p}^{GN}$ (radius cukup besar, percaya penuh Gauss–Newton).
2. Jika tidak, dan $\|\boldsymbol{p}^C\| \ge \Delta_k$: ambil $\boldsymbol{p}_k = (\Delta_k / \|\boldsymbol{p}^C\|)\, \boldsymbol{p}^C$ dengan $\boldsymbol{p}^C$ dari (7.10) (bahkan titik Cauchy pun sudah di luar radius; potong sepanjang arah penurunan tercuram).
3. Jika $\|\boldsymbol{p}^C\| < \Delta_k < \|\boldsymbol{p}^{GN}\|$, cari akar positif $\tau \in [0, 1]$ dari persamaan $\|\boldsymbol{p}^C + \tau(\boldsymbol{p}^{GN} - \boldsymbol{p}^C)\| = \Delta_k$. Ambil $\boldsymbol{p}_k = \boldsymbol{p}^C + \tau(\boldsymbol{p}^{GN} - \boldsymbol{p}^C)$.

Perbandingan kedua lintasan diperlihatkan pada Gambar 7.3 untuk data pada kotak kode berikut, yaitu J dan $\boldsymbol{r}_k$ yang sama tetapi dengan radius $\Delta_k = 1$.

Pada radius ini $\|\boldsymbol{p}^C\| \approx 0.85 < \Delta_k < \|\boldsymbol{p}^{GN}\| \approx 1.12$, sehingga yang berlaku adalah langkah 3 algoritma. Lintasan dogleg memotong lingkaran di dekat langkah eksak yang memenuhi $\|\boldsymbol{p}(\lambda^*)\| = \Delta_k$. Pada kotak kode, radius yang dipakai adalah $\Delta_k = 0.5$. Karena $\|\boldsymbol{p}^C\| \geq \Delta_k$, yang berlaku adalah langkah 2, dan hasilnya adalah titik Cauchy yang dipotong pada batas lingkaran.



Gambar 7.3: Lintasan *dogleg* $\mathbf{0} \to \boldsymbol{p}^C \to \boldsymbol{p}^{GN}$ dibandingkan dengan lintasan eksak $\boldsymbol{p}(\lambda)$ di dalam daerah kepercayaan $\|\boldsymbol{p}\| \leq \Delta_k$. Kontur menyatakan model kuadrat $\phi(\boldsymbol{p})$

Kode MATLAB: Langkah Dogleg Lengkap

```
clear; clc;
J = [1, 0; 0, 2; 1, 1];
r = [1; -1; 0.5];
g = J'*r;
pC = -(g'*g)/(g'*(J'*J)*g) * g;
pGN = -(J'*J) \ g;
Delta = 0.5;
if norm(pGN) <= Delta
    p = pGN;
elseif norm(pC) >= Delta
```

Kode MATLAB: Langkah Dogleg Lengkap *(lanjutan)*

```
    p = (Delta/norm(pC)) * pC;
else
    d = pGN - pC;
    a = d'*d; b = 2*pC'*d; c = pC'*pC - Delta^2;
    tau = (-b + sqrt(b^2 - 4*a*c)) / (2*a);
    p = pC + tau*d;
end
disp(p)
```

Karena tidak ada sistem linear baru yang harus diselesaikan untuk tiap Δ_k yang dicoba (cukup satu kali menghitung $\boldsymbol{p}^C$ dan $\boldsymbol{p}^{GN}$ di awal), dogleg jauh lebih murah daripada mencoba beberapa nilai λ berturut-turut. Konsekuensinya, lintasan patah-patah ini hanya *mendekati* lintasan $\boldsymbol{p}(\lambda)$ sesungguhnya, bukan menyamainya persis.

Persamaan Sekuler Moré: Pendekatan Eksak. Jika kita justru ingin $\|\boldsymbol{p}_k\| = \Delta_k$ *tepat* (bukan hanya perkiraan dogleg), kita perlu mencari λ sedemikian hingga $\|\boldsymbol{p}(\lambda)\| = \Delta_k$. Inilah **persamaan sekuler** (Moré, 1978; Moré dan Sorensen, 1983). Lemma 1 menjamin penyelesaiannya tunggal: karena $\|\boldsymbol{p}(\lambda)\|$ menurun monoton dari $\|\boldsymbol{p}^{GN}\|$ (di $\lambda = 0$) menuju 0 (saat $\lambda \to \infty$), dan kita sedang mencari kasus $\|\boldsymbol{p}^{GN}\| > \Delta_k$ (radius terlalu kecil untuk langkah Gauss–Newton), teorema nilai antara menjamin tepat satu $\lambda^* > 0$ dengan $\|\boldsymbol{p}(\lambda^*)\| = \Delta_k$.

Karena $\|\boldsymbol{p}(\lambda)\|$ dan turunannya dapat dihitung langsung dari SVD $J_k = U\Sigma V^\top$ melalui (7.6), nilai λ^* dapat dicari dengan metode Newton satu peubah. Kali ini metode Newton diterapkan pada fungsi skalar $\varphi(\lambda) := \|\boldsymbol{p}(\lambda)\| - \Delta_k$, bukan pada $S(\boldsymbol{\theta})$.

$$
\begin{aligned}
\lambda_{\text{baru}} &= \lambda - \frac{\varphi(\lambda)}{\varphi'(\lambda)}, \\
\varphi'(\lambda) &= -\frac{1}{\|\boldsymbol{p}(\lambda)\|} \sum_{i=1}^{n} \frac{c_i^2}{(\sigma_i^2 + \lambda)^3}, \\
c_i &:= \sigma_i \left[U^\top \boldsymbol{r}_k \right]_i .
\end{aligned}
\tag{7.11}
$$

Kode MATLAB: Metode Newton pada Persamaan Sekuler

```
clear; clc;
J = [1, 0; 0, 2; 1, 1];
r = [1; -1; 0.5];
Delta = 0.5;
[U, Sigma, V] = svd(J, 'econ');
sigma = diag(Sigma);
c = sigma .* (U'*r);
normp = @(lam) sqrt(sum((c./(sigma.^2+lam)).^2));

lambda = 1;
for it = 1:20
    np_ = normp(lambda);
    phi  = np_ - Delta;
    dphi = -sum(c.^2 ./ (sigma.^2+lambda).^3) / np_;
    lambda = max(lambda - phi/dphi, 1e-12);
    if abs(phi) < 1e-10, break; end
end
p = -V * (c./(sigma.^2+lambda));
disp(lambda)
disp(p)
```

Dalam praktik, iterasi Newton (7.11) pada $\varphi(\lambda)$ biasanya selesai dalam sedikit langkah. Metode ini dipakai ketika radius Δ_k perlu dipenuhi dengan teliti. Dogleg lebih sesuai ketika biaya komputasi per iterasi menjadi pertimbangan utama. Strategi Marquardt dan Nielsen berada di antara keduanya. Keduanya tidak membutuhkan SVD atau pencarian akar, tetapi masih memerlukan penyelesaian sistem linear untuk setiap percobaan λ.

Bab 8

Generalisasi ke Kuadrat Terkecil Berbobot

Sejauh ini, seluruh residu diperlakukan seolah-olah memiliki tingkat keandalan yang sama. Anggapan itu tidak selalu tepat. Galat pengukuran dapat memiliki variansi yang berbeda antarpengamatan. Dalam keadaan tersebut, pengamatan yang lebih teliti semestinya memberi pengaruh lebih besar.

Tingkat kepercayaan itu dinyatakan melalui matriks bobot $W \in \mathbb{R}^{m \times m}$ yang simetris dan definit positif. Bentuk yang paling umum ialah $W = \operatorname{diag}(w_1, \dots, w_m)$ dengan $w_i = 1/\sigma_i^2$. Fungsi objektifnya menjadi

$$S_W(\boldsymbol{\theta}) = \frac{1}{2} \boldsymbol{r}(\boldsymbol{\theta})^{\top} W \boldsymbol{r}(\boldsymbol{\theta}). \tag{8.1}$$

Bab ini menunjukkan bahwa perubahan ke (8.1) tidak memerlukan algoritma baru. Setelah residu dan Jacobian diskalakan dengan benar, seluruh perangkat Gauss–Newton tetap dapat digunakan.

8.1 Mengapa Data Tidak Selalu Boleh Diperlakukan Sama?

Bayangkan dua pengukuran untuk besaran yang sama. Pengukuran pertama berasal dari alat yang teliti, sedangkan pengukuran kedua berasal dari alat yang memiliki galat lebih besar. Jika keduanya diberi pengaruh sama, model akan berusaha menyesuaikan diri pada pengamatan yang kurang dapat dipercaya. Bobot memberi cara yang masuk akal untuk menyatakan tingkat kepercayaan tersebut.

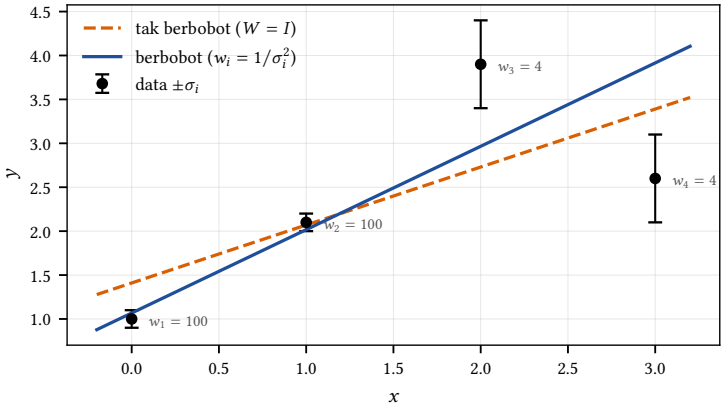
Untuk melihatnya secara konkret, anggap empat pengamatan memiliki simpangan baku $\sigma = (0.1, 0.1, 0.5, 0.5)$. Karena $w_i = 1/\sigma_i^2$, bobotnya ialah $(100, 100, 4, 4)$. Pengamatan pertama dan kedua masing-masing memperoleh bobot 25 kali lebih besar daripada pengamatan ketiga dan keempat. Penyebabnya bukan jumlah data, melainkan ketidakpastian yang lebih kecil. Nilai tersebut dirangkum pada Tabel 8.1.

Tabel 8.1: Contoh bobot dari simpangan baku pengukuran

Pengamatan i	1	2	3	4
σ_i	0.1	0.1	0.5	0.5
$w_i = 1/\sigma_i^2$	100	100	4	4

Contoh pada Tabel 8.1 menunjukkan satu hal penting. Bobot bukan angka tambahan yang ditempelkan setelah metode selesai. Bobot mengubah fungsi objektif sejak awal, sehingga residu, gradien, Jacobian efektif, dan arah langkah semuanya ikut berubah.

Pengaruh bobot pada hasil pencocokan dapat dilihat pada Gambar 8.1. Empat pengamatan $(x_i, y_i) = (0, 1.0), (1, 2.1), (2, 3.9)$, dan $(3, 2.6)$ diberi simpangan baku seperti pada Tabel 8.1. Garis tak berbobot $y \approx 1.41 + 0.66x$ tertarik oleh dua pengamatan terakhir yang kurang teliti. Garis berbobot $y \approx 1.07 + 0.95x$ melewati sangat dekat dua pengamatan pertama yang lebih dipercaya.



Gambar 8.1: Garis kuadrat terkecil tak berbobot dan berbobot pada data dengan ketidakpastian pengukuran yang berbeda

8.2 Dari Residu Asli ke Residu Efektif

Jika W diagonal dan definit positif, kita dapat menulis $W^{1/2} = \operatorname{diag}(\sqrt{w_1}, \dots, \sqrt{w_m})$. Dengan demikian,

$$S_W(\boldsymbol{\theta}) = \frac{1}{2} \left\| W^{1/2} \boldsymbol{r}(\boldsymbol{\theta}) \right\|^2 . \tag{8.2}$$

Artinya, berdasarkan (8.2), masalah berbobot tidak memerlukan teori optimisasi yang sepenuhnya baru. Kita cukup mengubah setiap residu menjadi $\widetilde{r}_i = \sqrt{w_i} r_i$ dan setiap baris Jacobian menjadi $\widetilde{J}_{i,:} = \sqrt{w_i} J_{i,:}$. Baris yang lebih dipercaya menjadi lebih besar, sedangkan baris yang kurang dipercaya menjadi lebih kecil.

Cara pandang ini juga membantu menghindari kesalahan umum. Jika bobot sudah dimasukkan ke residu efektif, jangan memasukkan W sekali lagi ketika membentuk persamaan normal. Bobot harus muncul tepat satu kali. Jika kita ulangi penurunan pada Bab 2–3 dengan W disisipkan pada tiap hasil kali dalam, gradien dan aproksimasi Hessian-nya menjadi

$$\nabla S_W(\boldsymbol{\theta}) = J(\boldsymbol{\theta})^\top W \boldsymbol{r}(\boldsymbol{\theta}), \qquad H_W(\boldsymbol{\theta}) \approx J(\boldsymbol{\theta})^\top W J(\boldsymbol{\theta}), \tag{8.3}$$

Dengan (8.3), persamaan normal Gauss–Newton berbobot berbentuk

$$\boxed{J_k^\top W J_k \, \boldsymbol{p}_k = -J_k^\top W \boldsymbol{r}_k} \tag{8.4}$$

Kode MATLAB: Kuadrat Terkecil Berbobot

```
clear; clc;
J = [1, 0; 1, 1; 1, 2; 1, 3];
r = [0.2; -0.1; 0.4; -0.2];
W = diag([4, 1, 9, 4]);
p = -(J'*W*J) \ (J'*W*r);
disp(p)
```

Catatan 4. *Persamaan* (8.4) *ini sebenarnya dapat direduksi kembali ke bentuk tak-berbobot* (3.13)*, melalui transformasi* $\tilde{\boldsymbol{r}} = W^{1/2}\boldsymbol{r}$ *dan* $\tilde{J} = W^{1/2}J$*. Hal ini berlaku karena* $\tilde{J}^\top \tilde{J} = J^\top W J$ *dan* $\tilde{J}^\top \tilde{\boldsymbol{r}} = J^\top W \boldsymbol{r}$*. Artinya, seluruh perangkat yang sudah kita bangun sejauh ini (SVD, QR, teorema konvergensi) langsung berlaku pada* $\tilde{J}$ *dan* $\tilde{\boldsymbol{r}}$*. Secara statistik, ketika* W *adalah kebalikan matriks kovariansi galat pengukuran, penyelesaian* (8.4) *bersesuaian dengan penaksir kemungkinan maksimum (*maximum likelihood estimator*) di bawah asumsi galat Gaussian.*

8.3 Langkah Perhitungan yang Dapat Diulang

Untuk menerapkan kuadrat terkecil berbobot pada data baru, ikuti urutan berikut.

1. Tentukan sumber ketidakpastian setiap pengamatan dan tuliskan simpangan bakunya σ_i.
2. Hitung $w_i = 1/\sigma_i^2$. Jika kovariansi antarpengamatan tidak nol, gunakan matriks kovariansi penuh dan ambil inversnya sebagai W.
3. Pada tebakan $\boldsymbol{\theta}_k$, hitung residu $\boldsymbol{r}_k$ dan Jacobian J_k.
4. Bentuk $\widetilde{\boldsymbol{r}}_k = W^{1/2}\boldsymbol{r}_k$ dan $\widetilde{J}_k = W^{1/2}J_k$.
5. Selesaikan $\widetilde{J}_k^\top \widetilde{J}_k \boldsymbol{p}_k = -\widetilde{J}_k^\top \widetilde{\boldsymbol{r}}_k$ melalui QR atau SVD.

6. Perbarui parameter, lalu bandingkan residu berbobot sebelum dan sesudah langkah.

Pelaporan hasil juga perlu menyebutkan apakah akurasi dihitung dari residu biasa atau residu berbobot. Nilai S_W yang kecil tidak otomatis berarti semua pengamatan cocok sama baiknya. Nilai tersebut berarti ketidakcocokan sudah dinilai sesuai tingkat kepercayaan pengukuran.

Bab 9

Contoh Numerik Lengkap

Sekarang mari kita lihat semua konsep di atas bekerja bersama-sama pada angka yang nyata. Tinjau model eksponensial $f(x, \theta) = \theta_1 e^{\theta_2 x}$ yang hendak dicocokkan dengan data $\{(x_i, y_i)\}$. Residunya adalah

$$r_i(\boldsymbol{\theta}) = y_i - \theta_1 e^{\theta_2 x_i}, \tag{9.1}$$

yang merupakan bentuk khusus (1.1), dengan turunan parsial (mengikuti definisi (2.2))

$$\frac{\partial r_i}{\partial \theta_1} = -e^{\theta_2 x_i}, \qquad \frac{\partial r_i}{\partial \theta_2} = -\theta_1 x_i e^{\theta_2 x_i}. \tag{9.2}$$

Data numerik. Misalkan kita memiliki $m = 4$ titik data yang tercantum pada Tabel 9.1. Tebakan awal yang dipakai adalah $\boldsymbol{\theta}_0 = (2,\ 0.6)$. Prediksi model

Tabel 9.1: Data numerik untuk regresi eksponensial

i	1	2	3	4
x_i	0	1	2	3
y_i	2.1	3.9	7.8	15.2

pada tebakan awal ini adalah $f(x_i, \boldsymbol{\theta}_0) = 2e^{0.6 x_i} = (2.000,\ 3.644,\ 6.640,\ 12.099)$,

sehingga vektor residunya menjadi

$$\boldsymbol{r}_0 = \begin{pmatrix} 2.1 - 2.000 \\ 3.9 - 3.644 \\ 7.8 - 6.640 \\ 15.2 - 12.099 \end{pmatrix} = \begin{pmatrix} 0.100 \\ 0.256 \\ 1.160 \\ 3.101 \end{pmatrix}. \tag{9.3}$$

Menurut (9.2), kolom pertama Jacobian adalah $-e^{0.6x_i}$ (sensitivitas terhadap θ_1), sedangkan kolom kedua adalah $-\theta_1 x_i e^{0.6x_i} = -2x_i e^{0.6x_i}$ (sensitivitas terhadap θ_2), mengikuti (2.1):

$$J_0 = \begin{pmatrix} -1.000 & 0.000 \\ -1.822 & -3.644 \\ -3.320 & -13.280 \\ -6.050 & -36.298 \end{pmatrix} \in \mathbb{R}^{4\times 2}. \tag{9.4}$$

Mengikuti formula Gram (3.8), matriks $J_0^\top J_0$ dihitung sebagai hasil kali dalam antarkolom J_0 pada (9.4):

$$\begin{aligned}
[J_0^\top J_0]_{11} &= (-1.000)^2 + (-1.822)^2 + (-3.320)^2 + (-6.050)^2 \\
&= 1.000 + 3.320 + 11.022 + 36.603 = 51.945, \\
[J_0^\top J_0]_{12} &= [J_0^\top J_0]_{21} \\
&= (-1.000)(0) + (-1.822)(-3.644) \\
&\quad + (-3.320)(-13.280) + (-6.050)(-36.298) \\
&= 0.000 + 6.640 + 44.090 + 219.603 = 270.333, \\
[J_0^\top J_0]_{22} &= 0^2 + (-3.644)^2 + (-13.280)^2 \\
&\quad + (-36.298)^2 \\
&= 0.000 + 13.279 + 176.358 \\
&\quad + 1317.545 = 1507.182,
\end{aligned}$$

sehingga $J_0^\top J_0 = \begin{pmatrix} 51.945 & 270.333 \\ 270.333 & 1507.182 \end{pmatrix}$. Matriks ini simetris, sesuai sifat matriks Gram pada Bab 3. Matriks tersebut juga definit positif karena determinannya $51.945 \times 1507.182 - 270.333^2 = 5210.6 > 0$.

Dengan cara yang sama, vektor $J_0^\top \boldsymbol{r}_0$ dihitung dari (9.4) dan (9.3):

$$\begin{aligned} [J_0^\top \boldsymbol{r}_0]_1 &= (-1.000)(0.100) + (-1.822)(0.256) \\ &\quad + (-3.320)(1.160) + (-6.050)(3.101) = -23.178, \\ [J_0^\top \boldsymbol{r}_0]_2 &= (0)(0.100) + (-3.644)(0.256) \\ &\quad + (-13.280)(1.160) + (-36.298)(3.101) = -128.887. \end{aligned}$$

Persamaan normal (3.13) pada iterasi ini menjadi sistem 2×2 berikut:

$$\begin{pmatrix} 51.945 & 270.333 \\ 270.333 & 1507.182 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} = \begin{pmatrix} 23.178 \\ 128.887 \end{pmatrix}, \tag{9.5}$$

Sistem (9.5) dapat diselesaikan dengan eliminasi Gauss (atau, untuk sistem sekecil ini, aturan Cramer), sehingga diperoleh $\boldsymbol{p}_0 \approx (0.017,\ 0.082)$. Parameter pun diperbarui menjadi $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_0 + \boldsymbol{p}_0 \approx (2.017,\ 0.682)$. Proses ini kemudian diulang: hitung $\boldsymbol{r}_1$ dan J_1, lalu selesaikan kembali persamaan normal sampai $\|J_k^\top \boldsymbol{r}_k\|$ cukup kecil.

Langkah Eksperimen

1. Masukkan data $(x, y) = \{(0, 2.1), (1, 3.9), (2, 7.8), (3, 15.2)\}$ dan pilih $f(x, a, b) = ae^{bx}$.
2. Tetapkan $S = \frac{1}{2}\sum_i r_i^2$ dan tebakan awal $(a, b) = (2, 0.6)$.
3. Hitung $r_i = y_i - ae^{bx_i}$ serta baris Jacobian $(-e^{bx_i}, -ax_i e^{bx_i})$ sesuai (9.1)–(9.2).
4. Selesaikan $J^\top J \boldsymbol{p} = -J^\top \boldsymbol{r}$, kemudian perbarui $(a, b) \leftarrow (a, b) + \boldsymbol{p}$.
5. Ulangi sampai $\|J^\top \boldsymbol{r}\| < 10^{-8}$ atau paling banyak 10 putaran.

Langkah Eksperimen *(lanjutan)*

6. Periksa bahwa hasil akhirnya mendekati $(a, b) = (2.0334, 0.6706)$ dengan $S = 0.0054335$.

9.1 Membaca Kurva Pembelajaran: Gauss–Newton versus Penurunan Gradien

Satu iterasi yang berhasil belum memberi gambaran lengkap tentang perilaku sebuah metode. Karena itu, kita bandingkan seluruh proses pembelajaran pada contoh eksponensial yang sama. Kedua metode dimulai dari $\boldsymbol{\theta}_0 = (2, 0.6)$ dan meminimumkan fungsi objektif yang sama. Gauss–Newton menyelesaikan subproblem kuadrat terkecil dengan dekomposisi SVD, sedangkan penurunan gradien memakai laju belajar tetap $\alpha = 5 \times 10^{-4}$.

Intuisi

Kurva pembelajaran menggambarkan kecepatan penurunan kesalahan. Kurva yang turun tajam berarti metode cepat memperbaiki model. Kurva yang mendatar berarti perbaikannya mulai kecil. Pada gambar berikut, sumbu vertikal menampilkan selisih $S(\boldsymbol{\theta}_k) - S^*$ terhadap nilai minimum dalam skala logaritmik, agar penurunan tajam dan penurunan lambat dapat terlihat dalam satu gambar.

Langkah Eksperimen Perbandingan Dua Metode

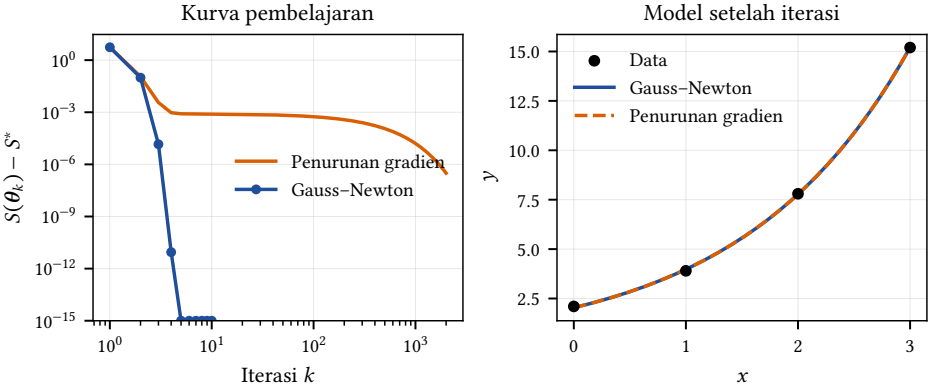
1. Gunakan data, model, dan tebakan awal dari contoh eksponensial sebelumnya. Hitung $\boldsymbol{r}(\boldsymbol{\theta}_k)$ dan $J(\boldsymbol{\theta}_k)$.
2. Untuk Gauss–Newton, bentuk $\nabla S = J^\top \boldsymbol{r}$ dan selesaikan (3.13) dengan SVD. Simpan $S(\boldsymbol{\theta}_k)$ sebelum setiap pembaruan dan jalankan 10 iterasi.
3. Untuk penurunan gradien, gunakan (3.16) dengan $\alpha = 5 \times 10^{-4}$ selama 2000 iterasi.
4. Hitung S^* sebagai nilai objektif akhir Gauss–Newton. Gambarkan selisih

Langkah Eksperimen Perbandingan Dua Metode *(lanjutan)*

$S(\boldsymbol{\theta}_k) - S^*$ kedua deret dengan sumbu logaritmik.

5. Periksa bahwa Gauss–Newton mencapai $S < 0.006$ sekitar iterasi ke-2, sedangkan penurunan gradien memerlukan sekitar 95 pembaruan.

Gambar 9.1 memperlihatkan kurva pembelajaran dan hasil pencocokan model eksponensial.



Gambar 9.1: Kurva pembelajaran (selisih terhadap nilai minimum S^*) dan hasil pencocokan model eksponensial

Hasil numeriknya diringkas pada Tabel 9.2.

Tabel 9.2: Hasil numerik regresi eksponensial

Metode	Iterasi	$\boldsymbol{\theta}$ akhir	S akhir
Gauss–Newton	10	(2.0334, 0.6706)	0.005433537
penurunan gradien	2000	(2.0330, 0.6706)	0.005433848

Pada contoh ini, Gauss–Newton mencapai $S < 0.006$ setelah dua pembaruan. Penurunan gradien memerlukan sekitar 95 pembaruan untuk mencapai tingkat kesalahan yang sama. Setelah cukup lama, keduanya menemukan parameter yang hampir sama karena mengoptimalkan model dan data yang sama.

Perbedaan utamanya terletak pada cara memilih arah dan panjang langkah. Gauss–Newton menggunakan informasi bentuk lokal melalui Jacobian, sedangkan penurunan gradien hanya mengikuti gradien dengan ukuran langkah tetap.

Perbandingan ini juga menunjukkan batas interpretasi kurva. Penurunan gradien dapat dibuat lebih cepat dengan memilih laju belajar yang lebih baik, tetapi laju yang terlalu besar dapat membuatnya tidak stabil. Sebaliknya, Gauss–Newton dapat mengambil langkah yang sangat efisien ketika model lokalnya akurat, tetapi tetap memerlukan regularisasi atau pencarian garis pada masalah yang lebih sulit.

9.2 Eksperimen Publik: SLP untuk Regresi Diabetes

Contoh eksponensial sebelumnya membantu kita menghitung Jacobian dengan tangan. Sekarang kita gunakan data publik yang lebih realistis. Dataset Diabetes yang didistribusikan melalui scikit-learn memiliki 442 pengamatan, 10 fitur, dan target perkembangan penyakit (Scikit-learn Developers, 2026). Untuk menjaga alur tetap mudah diikuti, eksperimen hanya memakai fitur indeks massa tubuh atau BMI.

Model ini termasuk SLP regresi karena keluarannya hanya satu unit dan tidak memiliki lapisan tersembunyi. Suku t^2 hanyalah fitur hasil transformasi. Dengan vektor masukan $\boldsymbol{u}_i = (1, t_i, t_i^2)$, model tetap linear terhadap parameternya, yaitu $f_i = \boldsymbol{u}_i^\top \boldsymbol{\theta}$.

Model yang dipakai adalah polinomial kuadrat pada BMI terstandardisasi t:

$$f(t, \boldsymbol{\theta}) = \theta_0 + \theta_1 t + \theta_2 t^2, \qquad r_i(\boldsymbol{\theta}) = z_i - f(t_i, \boldsymbol{\theta}). \tag{9.6}$$

Dengan $\boldsymbol{u}_i = (1, t_i, t_i^2)$, baris Jacobian residunya langsung mengikuti (2.2):

$$J_{i,:} = (-1, -t_i, -t_i^2), \qquad \nabla S = J^\top \boldsymbol{r}. \tag{9.7}$$

Langkah Eksperimen

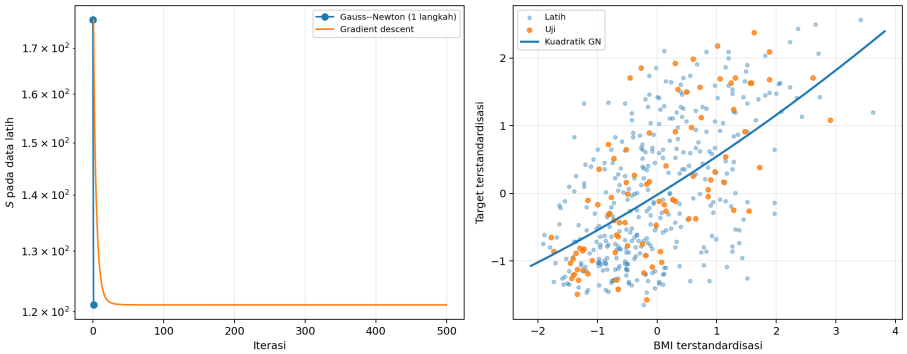
Lakukan tahapan berikut sebelum membuat gambar dan tabel.

1. Muat dataset `load_diabetes` yang didokumentasikan oleh Scikit-learn Developers (2026). Ambil kolom ketiga sebagai BMI dan target perkembangan penyakit sebagai y. Dataset tersebut memiliki 442 baris.
2. Acak indeks data menggunakan generator bilangan acak dengan *seed* 42. Pada eksperimen acuan, generator yang dipakai adalah `numpy.random.default_rng(42)` dengan fungsi `permutation`. Ambil 353 baris pertama untuk latih dan 89 baris sisanya untuk uji.
3. Hitung rerata dan simpangan baku BMI serta target hanya dari data latih. Ubah keduanya menjadi variabel terstandardisasi t dan z. Pada eksperimen acuan, rerata target adalah 150.6374 dan simpangan bakunya 76.2086.
4. Bentuk f, $\boldsymbol{r}$, dan J dari (9.6)–(9.7). Mulai dari $\boldsymbol{\theta}_0 = \mathbf{0}$.
5. Untuk Gauss–Newton, selesaikan (3.13) dengan SVD, lalu perbarui parameter satu kali. Untuk penurunan gradien, gunakan (3.16) dalam bentuk ternormalisasi $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - (0.12/353)J^{\top}\boldsymbol{r}$ selama 500 iterasi.
6. Simpan $S = \frac{1}{2}\|\boldsymbol{r}\|^2$ pada data latih. Kembalikan prediksi ke satuan target semula, lalu hitung RMSE pada 89 data uji. Setelah itu, gambarkan kurva S dan garis model kuadratik.

Kurva pembelajaran ditampilkan pada Gambar 9.2, sedangkan hasil numeriknya dirangkum dalam Tabel 9.3. Sumbu target pada grafik masih memakai satuan terstandardisasi agar kedua metode dapat dibandingkan pada skala yang sama.

Tabel 9.3 merangkum hasil numeriknya.

Satu langkah Gauss–Newton langsung membawa model ke solusi kuadrat terkecil. Penurunan gradien akhirnya mencapai titik yang sama, tetapi memer-



Gambar 9.2: Kurva pembelajaran dan pencocokan regresi kuadratik pada data Diabetes

Tabel 9.3: Hasil SLP untuk regresi Diabetes

Metode	Iterasi	S akhir	RMSE uji
Gauss–Newton	1	121.0833	59.664
Penurunan gradien ($\alpha = 0.12/353$)	500	121.0833	59.664

lukan banyak pembaruan. Contoh ini memperlihatkan bahwa data publik tetap dapat digunakan untuk memahami persamaan dasar tanpa membuat model menjadi terlalu rumit.

Bab 10

Penerapan pada Klasifikasi Biner: Regresi Logistik dan IRLS

Sejauh ini kita membahas regresi, yaitu mencocokkan model dengan data numerik kontinu $y_i \in \mathbb{R}$. Sekarang kita beralih ke situasi ketika y_i merupakan label kelas. Contohnya adalah $y_i \in \{0, 1\}$ untuk menyatakan "gagal" atau "lulus", serta "spam" atau "bukan spam". Gagasan utama Gauss–Newton masih dapat digunakan. Kita hanya perlu menyesuaikan fungsi kerugiannya. Penyesuaian tersebut kita bangun langkah demi langkah.

10.1 Mengapa Kuadrat Terkecil Tidak Lagi Tepat

Pada masalah klasifikasi biner, keluaran model bukan lagi sembarang bilangan real, melainkan sebuah *peluang* bahwa $y_i = 1$. Cara paling umum untuk memastikan keluaran model selalu berada di rentang $(0, 1)$ adalah memakai fungsi **sigmoid** $\sigma(z) = 1/(1 + e^{-z})$ pada sebuah kombinasi linear dari fitur:

$$\pi_i(\boldsymbol{\theta}) = \sigma\left(\boldsymbol{x}_i^\top \boldsymbol{\theta}\right) = \frac{1}{1 + e^{-\boldsymbol{x}_i^\top \boldsymbol{\theta}}}, \qquad i = 1, \dots, m, \tag{10.1}$$

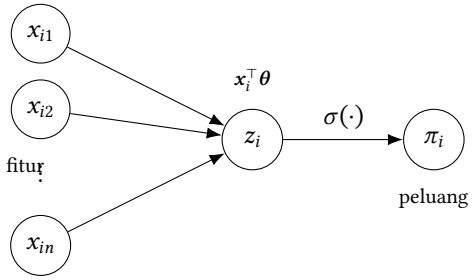
dengan $\boldsymbol{x}_i \in \mathbb{R}^n$ sebagai vektor fitur pengamatan ke-i (biasanya memuat entri konstan 1 untuk suku intersep) dan $\boldsymbol{\theta} \in \mathbb{R}^n$ sebagai parameter (bobot) yang ingin

kita cari. Model ini disebut perceptron satu lapisan: seluruh fitur terhubung *langsung* ke satu unit keluaran, tanpa lapisan tersembunyi di antaranya. Model ini berbeda dengan MLP yang dibahas pada Bab 13–14.

Kode MATLAB: Model Sigmoid

```
clear; clc;
X = [1, -2; 1, -1; 1, 1; 1, 2];
theta = [0.2; 1.0];
z   = X * theta;
pi_ = 1 ./ (1 + exp(-z));
disp(pi_)
```

Hubungan langsung antara fitur, logit, dan peluang pada model ini diperlihatkan pada Gambar 10.1.



Gambar 10.1: Arsitektur perceptron satu lapisan untuk klasifikasi biner

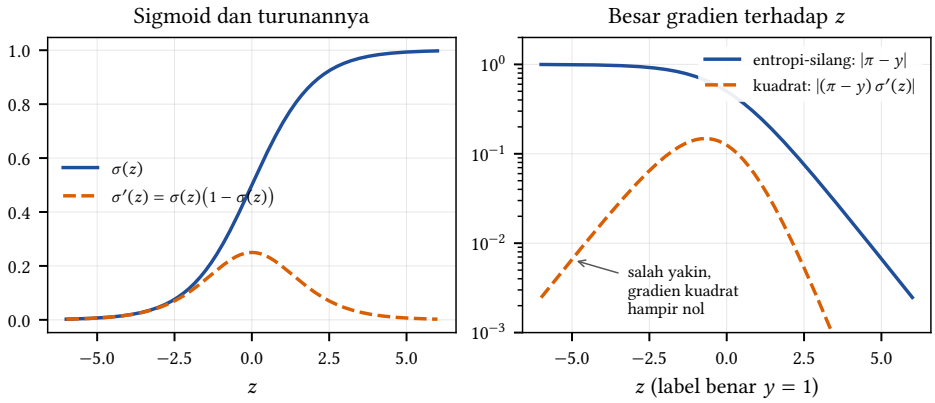
Kita mungkin tergoda memakai fungsi kuadrat yang sudah dikenal, yaitu $S(\boldsymbol{\theta}) = \frac{1}{2}\sum_i (y_i - \pi_i(\boldsymbol{\theta}))^2$. Pilihan ini memiliki kelemahan serius. Saat z jauh dari nol, sigmoid $\sigma(z)$ mendatar dan turunannya mendekati nol. Akibatnya, gradien S juga kecil pada titik yang salah diklasifikasikan dengan yakin. Model menjadi lambat belajar dari kesalahannya sendiri (Bishop, 2006). Fungsi yang lebih sesuai dan berkaitan langsung dengan estimasi kemungkinan maksimum adalah **kerugian entropi-silang** (*cross-entropy*):

$$L(\boldsymbol{\theta}) = -\sum_{i=1}^{m} \Big[y_i \log \pi_i(\boldsymbol{\theta}) + (1 - y_i) \log \big(1 - \pi_i(\boldsymbol{\theta})\big) \Big]. \tag{10.2}$$

Kode MATLAB: Kerugian Entropi-Silang

```
clear; clc;
X = [1, -2; 1, -1; 1, 1; 1, 2];
y = [0; 0; 1; 1];
theta = [0.2; 1.0];
z = X * theta;
pi_ = 1 ./ (1 + exp(-z));
L = -sum( y.*log(pi_) + (1-y).*log(1-pi_) );
disp(L)
```

Alasan mengapa entropi-silang lebih sesuai terlihat pada Gambar 10.2. Untuk data berlabel $y = 1$, turunan kerugian kuadrat terhadap logit mengandung faktor $\sigma'(z)$, sehingga besarnya hampir nol ketika z sangat negatif. Keadaan ini justru terjadi saat model salah dengan yakin. Turunan entropi-silang, yaitu $\pi - y$, tetap mendekati 1 pada keadaan tersebut. Rumus kedua turunan ini diturunkan pada subbab berikutnya.



Gambar 10.2: Kiri: fungsi sigmoid dan turunannya. Kanan: besar gradien terhadap logit z untuk kerugian entropi-silang dan kerugian kuadrat ketika label benar $y = 1$

Fungsi $L(\boldsymbol{\theta})$ pada (10.2) tidak lagi berbentuk $\frac{1}{2}\sum_i r_i^2$. Karena itu, perangkat pada Bab 3 tidak dapat dipakai secara langsung. Namun, gagasan dasarnya tetap sama. Kita akan menurunkan gradien dan Hessian L dengan aturan rantai, lalu melihat struktur yang menyerupai persamaan normal.

10.2 Turunan Pertama dan Kedua dari Kerugian Entropi-Silang

Sebelum menurunkan gradien dan Hessian $L(\boldsymbol{\theta})$, kita perlu menghitung turunan sigmoid. Tulis $\sigma(z) = (1+e^{-z})^{-1}$. Aturan pangkat dan aturan rantai memberikan

$$\begin{aligned}\sigma'(z) &= -(1+e^{-z})^{-2}\frac{d}{dz}(1+e^{-z}) \\ &= -(1+e^{-z})^{-2}(-e^{-z}) = \frac{e^{-z}}{(1+e^{-z})^2} \\ &= \frac{1}{1+e^{-z}}\left(\frac{e^{-z}}{1+e^{-z}}\right) = \sigma(z)\big(1-\sigma(z)\big). \end{aligned} \tag{10.3}$$

Kesetaraan terakhir memakai $1-\sigma(z) = e^{-z}/(1+e^{-z})$. Sekarang ambil satu suku kerugian

$$L_i = -y_i \log \pi_i - (1-y_i)\log(1-\pi_i), \qquad \pi_i = \sigma(z_i), \qquad z_i = \boldsymbol{x}_i^{\top}\boldsymbol{\theta}.$$

Kita turunkan dahulu L_i terhadap logit z_i, bukan langsung terhadap $\boldsymbol{\theta}$. Pemisahan ini membuat aturan rantai lebih mudah diikuti. Turunan suku pertama memakai $d[\log u]/dz = u'/u$. Pada suku kedua muncul satu tanda negatif tambahan karena $d(1-\pi_i)/dz_i = -\pi_i'$. Dengan memakai (10.3), diperoleh

$$\begin{aligned}\frac{\partial L_i}{\partial z_i} &= -y_i\frac{1}{\pi_i}\frac{\partial \pi_i}{\partial z_i} - (1-y_i)\frac{1}{1-\pi_i}\left(-\frac{\partial \pi_i}{\partial z_i}\right) \\ &= -y_i\frac{\pi_i(1-\pi_i)}{\pi_i} + (1-y_i)\frac{\pi_i(1-\pi_i)}{1-\pi_i} \\ &= -y_i(1-\pi_i) + (1-y_i)\pi_i \\ &= -y_i + y_i\pi_i + \pi_i - y_i\pi_i = \pi_i - y_i. \end{aligned} \tag{10.4}$$

Karena y_i adalah label data yang konstan terhadap parameter, turunan keduanya menjadi

$$\frac{\partial^2 L_i}{\partial z_i^2} = \frac{\partial}{\partial z_i}(\pi_i - y_i) = \frac{\partial \pi_i}{\partial z_i} = \pi_i(1-\pi_i). \tag{10.5}$$

Kedua hasil (10.4) dan (10.5) berlaku per pengamatan. Karena $L = \sum_i L_i$, gradien keseluruhan diperoleh dengan menjumlahkan kontribusi seluruh pengamatan. Keduanya sederhana dan pantas diingat: turunan pertama kerugian tidak lain adalah selisih antara peluang prediksi dan label sebenarnya, $\pi_i - y_i$. Bentuk ini sama persis dengan residu $-r_i$ pada kasus regresi. Turunan keduanya, $\pi_i(1-\pi_i)$, selalu bernilai di antara 0 dan $\frac{1}{4}$, dan mengecil justru ketika model sangat yakin (π_i dekat 0 atau 1).

10.3 Menurunkan Metode Newton untuk Regresi Logistik

Sekarang kita terapkan aturan rantai sekali lagi, kali ini dari z_i ke $\boldsymbol{\theta}$. Karena $z_i(\boldsymbol{\theta}) = \boldsymbol{x}_i^\top \boldsymbol{\theta}$ adalah fungsi *linear* terhadap $\boldsymbol{\theta}$, turunannya terhadap $\boldsymbol{\theta}$ sederhana: $\partial z_i / \partial \theta_l = x_{il}$, sehingga Jacobian dari $\boldsymbol{z}(\boldsymbol{\theta}) = (z_1, \dots, z_m)$ terhadap $\boldsymbol{\theta}$ tidak lain adalah matriks desain itu sendiri,

$$J = X = \begin{pmatrix} \boldsymbol{x}_1^\top \\ \vdots \\ \boldsymbol{x}_m^\top \end{pmatrix} \in \mathbb{R}^{m \times n},$$

yang berperan persis seperti matriks Jacobian pada Bab 2. Bedanya, di sini matriks tersebut *konstan* dan tidak berubah terhadap $\boldsymbol{\theta}$ karena model logitnya sudah linear. Dengan aturan rantai dan hasil di atas, gradien $L(\boldsymbol{\theta})$ menjadi

$$\nabla L(\boldsymbol{\theta}) = \sum_{i=1}^{m} \frac{\partial L}{\partial z_i} \frac{\partial z_i}{\partial \boldsymbol{\theta}} = X^\top \big(\boldsymbol{\pi}(\boldsymbol{\theta}) - \boldsymbol{y} \big), \tag{10.6}$$

dengan $\boldsymbol{\pi}(\boldsymbol{\theta}) = (\pi_1(\boldsymbol{\theta}), \dots, \pi_m(\boldsymbol{\theta}))^\top$. Bandingkan bentuk ini dengan $\nabla S(\boldsymbol{\theta}) = J^\top \boldsymbol{r}(\boldsymbol{\theta})$ pada (2.4). Strukturnya sama persis, hanya saja "residu" $\boldsymbol{r}$ digantikan oleh "selisih peluang" $\boldsymbol{\pi} - \boldsymbol{y}$.

Kode MATLAB: Gradien Regresi Logistik

```
clear; clc;
X = [1, -2; 1, -1; 1, 1; 1, 2];
y = [0; 0; 1; 1];
theta = [0.2; 1.0];
z = X * theta;
pi_ = 1 ./ (1 + exp(-z));
grad = X' * (pi_ - y);
disp(grad)
```

Sekarang tiba bagian yang paling menarik. Mari kita hitung Hessian eksaknya, mengikuti pola dekomposisi yang sama seperti (3.5) di Bab 3 (turunkan gradien sekali lagi, pakai aturan hasil kali):

$$\nabla^2 L(\boldsymbol{\theta}) = X^\top D(\boldsymbol{\theta}) X + \sum_{i=1}^{m} \big(\pi_i(\boldsymbol{\theta}) - y_i\big) \nabla^2 z_i(\boldsymbol{\theta}), \tag{10.7}$$

$$D(\boldsymbol{\theta}) := \operatorname{diag}\big(\pi_1(1-\pi_1), \dots, \pi_m(1-\pi_m)\big).$$

Catatan Aljabar Linear: Mengapa Aproksimasi Gauss–Newton di Sini Bukan Aproksimasi Sama Sekali

Suku kedua pada (10.7) merupakan padanan $M(\boldsymbol{\theta}) = \sum_i r_i \nabla^2 r_i$ pada Bab 3. Di sini $z_i(\boldsymbol{\theta}) = \boldsymbol{x}_i^\top \boldsymbol{\theta}$ linear terhadap parameter. Karena itu, $\nabla^2 z_i(\boldsymbol{\theta}) \equiv \mathbf{0}$ untuk setiap $\boldsymbol{\theta}$, bukan hanya di sekitar solusi. Suku kedua pun lenyap dan

$$\nabla^2 L(\boldsymbol{\theta}) = X^\top D(\boldsymbol{\theta}) X \quad \text{secara } \textit{eksak}\text{, bukan aproksimasi.}$$

Bandingkan dengan Teorema 1 di Bab 6. Untuk regresi logistik, $M(\boldsymbol{\theta}) \equiv \mathbf{0}$, sehingga langkah Newton dan Gauss–Newton berimpit. Jika terdapat titik minimum berhingga dan $X^\top D(\boldsymbol{\theta}) X$ tetap definit positif di sekitarnya, konvergensi lokalnya bersifat kuadratik. Syarat tentang titik minimum berhingga penting, sebagaimana akan terlihat pada contoh data terpisah sempurna.

10.4 Persamaan Newton sebagai Kuadrat Terkecil Berbobot: Algoritma IRLS

Karena Hessian-nya persis $X^\top D(\boldsymbol{\theta})X$, langkah Newton $\boldsymbol{p}_k = -\nabla^2 L(\boldsymbol{\theta}_k)^{-1}\nabla L(\boldsymbol{\theta}_k)$ pada iterasi ke-k memenuhi

$$\boxed{X^\top D_k X\, \boldsymbol{p}_k = -X^\top\big(\boldsymbol{\pi}_k - \boldsymbol{y}\big)}, \qquad D_k := D(\boldsymbol{\theta}_k), \quad \boldsymbol{\pi}_k := \boldsymbol{\pi}(\boldsymbol{\theta}_k). \tag{10.8}$$

Kode MATLAB: Satu Iterasi IRLS

```
clear; clc;
X = [1, -2; 1, -1; 1, 1; 1, 2];
y = [0; 0; 1; 1];
theta = [0; 0];
z    = X * theta;
pi_  = 1 ./ (1 + exp(-z));
D    = diag( pi_ .* (1 - pi_) );
p    = -(X'*D*X) \ (X'*(pi_ - y));
theta = theta + p;
disp(theta)
```

Persamaan ini identik dengan persamaan normal berbobot (8.4), dengan $J = X$ dan $W = D_k$. Perbedaannya terletak pada bobot. Pada Bab 8, W tetap. Dalam regresi logistik, D_k berubah pada setiap iterasi karena bergantung pada $\boldsymbol{\pi}_k = \boldsymbol{\pi}(\boldsymbol{\theta}_k)$. Karena itu, algoritma ini disebut IRLS (Green, 1984; McCullagh dan Nelder, 1989) (*Iteratively Reweighted Least Squares*). Setiap iterasi menyelesaikan satu masalah kuadrat terkecil berbobot, memperbarui parameter, kemudian menghitung ulang bobot.

Algoritma: IRLS untuk Regresi Logistik

1. Berikan tebakan awal $\boldsymbol{\theta}_0$ (misalnya $\boldsymbol{\theta}_0 = \mathbf{0}$), toleransi $\varepsilon > 0$, dan set $k = 0$.

2. Hitung $\boldsymbol{z}_k = X\boldsymbol{\theta}_k$, lalu $\boldsymbol{\pi}_k = \sigma(\boldsymbol{z}_k)$ elemen-demi-elemen.

Prosedur (lanjutan)

3. Bentuk matriks bobot $D_k = \operatorname{diag}\big(\pi_{k,i}(1-\pi_{k,i})\big)$.
4. Selesaikan sistem linear $X^\top D_k X\, \boldsymbol{p}_k = -X^\top(\boldsymbol{\pi}_k - \boldsymbol{y})$ (melalui QR/SVD seperti Bab 4 bila $X^\top D_k X$ mendekati singular).
5. Perbarui: $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{p}_k$.
6. Jika $\|X^\top(\boldsymbol{\pi}_k - \boldsymbol{y})\| < \varepsilon$, berhenti. Jika belum, $k \leftarrow k+1$ dan kembali ke langkah 2.

Intuisi

IRLS dapat dibaca sebagai siklus sederhana: (1) hitung peluang dengan parameter saat ini, (2) beri bobot lebih besar pada titik yang informasinya lebih kuat, (3) selesaikan masalah kuadrat terkecil berbobot, lalu (4) perbarui parameter. Siklus ini diulang sampai gradiennya kecil.

10.5 Contoh Numerik Singkat

Tinjau $m = 4$ pengamatan dengan satu fitur (plus intersep), $\boldsymbol{x}_i = (1, x_i)$, dan label biner seperti pada Tabel 10.1. Matriks desainnya adalah $X = \begin{pmatrix} 1 & -2 \\ 1 & -1 \\ 1 & 1 \\ 1 & 2 \end{pmatrix}$, dan

Tabel 10.1: Contoh data klasifikasi biner

i	1	2	3	4
x_i	-2	-1	1	2
y_i	0	0	1	1

tebakan awal $\boldsymbol{\theta}_0 = (0, 0)$ (artinya belum ada informasi apa pun, semua logit nol).

Karena $\boldsymbol{\theta}_0 = \mathbf{0}$, seluruh logit $z_i = \boldsymbol{x}_i^\top \boldsymbol{\theta}_0 = 0$, sehingga $\pi_i = \sigma(0) = 0.5$ untuk keempat titik. Selisih peluangnya adalah $\boldsymbol{\pi}_0 - \boldsymbol{y} = (0.5,\, 0.5,\, -0.5,\, -0.5)$.

Komponen gradien pada (10.6) adalah

$$\begin{aligned}[X^\top(\boldsymbol{\pi}_0 - \boldsymbol{y})]_1 &= 0.5 + 0.5 - 0.5 - 0.5 = 0,\\ [X^\top(\boldsymbol{\pi}_0 - \boldsymbol{y})]_2 &= (-2)(0.5) + (-1)(0.5)\\ &\quad + (1)(-0.5) + (2)(-0.5) = -3.\end{aligned}$$

Karena $\pi_i = 0.5$ untuk semua i, bobotnya seragam: $D_0 = 0.25\, I_4$. Dengan demikian, $X^\top D_0 X = 0.25\, X^\top X$. Karena data ini simetris terhadap nol, diperoleh

$$X^\top X = \begin{pmatrix} 4 & 0 \\ 0 & 10 \end{pmatrix} \quad\Longrightarrow\quad X^\top D_0 X = \begin{pmatrix} 1 & 0 \\ 0 & 2.5 \end{pmatrix}.$$

Persamaan normal IRLS (10.8) pada iterasi ini menjadi

$$\begin{pmatrix} 1 & 0 \\ 0 & 2.5 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \end{pmatrix} \quad\Longrightarrow\quad \boldsymbol{p}_0 = (0,\ 1.2). \tag{10.9}$$

Jadi, dari (10.9), $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_0 + \boldsymbol{p}_0 = (0, 1.2)$. Intersep tetap nol karena data seimbang dan simetris di sekitar $x = 0$. Koefisien kemiringan naik dari 0 menjadi 1.2, sehingga model mulai mengaitkan nilai x yang lebih besar dengan kelas $y = 1$.

Contoh ini memiliki satu keistimewaan yang tidak boleh diabaikan. Kedua kelas terpisah sempurna oleh batas $x = 0$. Akibatnya, regresi logistik tanpa regularisasi tidak memiliki penaksir kemungkinan maksimum yang berhingga. Iterasi IRLS terus memperbesar kemiringan sambil menurunkan kerugian. Karena itu, pembahasan konvergensi kuadratik di sekitar titik minimum berhingga tidak berlaku pada data kecil ini.

Langkah Eksperimen

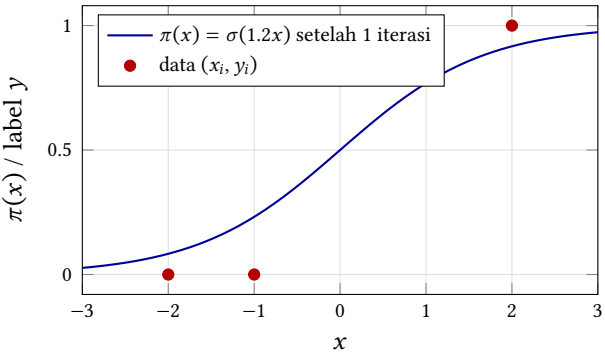
Sebelum membuat gambar, ikuti urutan berikut.

1. Masukkan empat pasangan $(x, y) = \{(-2, 0), (-1, 0), (1, 1), (2, 1)\}$. Bentuk $X = [\mathbf{1}\ \boldsymbol{x}]$ dan pilih $\boldsymbol{\theta}_0 = (0, 0)$.
2. Hitung peluang $\boldsymbol{\pi}$ dengan (10.1). Bentuk $D = \mathrm{diag}(\pi_i(1 - \pi_i))$.

Langkah Eksperimen *(lanjutan)*

3. Selesaikan $X^\top D X \boldsymbol{p} = -X^\top(\boldsymbol{\pi} - \boldsymbol{y})$ sesuai (10.8), lalu hitung $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_0 + \boldsymbol{p}$.
4. Periksa bahwa langkah pertama memberi $\boldsymbol{p} = (0, 1.2)$. Gunakan parameter tersebut untuk menggambar $\pi(x) = \sigma(1.2x)$.
5. Jika iterasi dilanjutkan, catat bahwa kemiringan terus membesar. Tambahkan regularisasi L2 jika tujuan eksperimen ialah memperoleh parameter akhir yang berhingga.

Agar apa yang sebenarnya terjadi lebih mudah dipahami, Gambar 10.3 menampilkan data dan kurva $\pi(x) = \sigma(\theta_1 + \theta_2 x)$ setelah satu iterasi.



Gambar 10.3: Kurva sigmoid setelah satu iterasi IRLS pada contoh klasifikasi biner

Kurva sigmoid ini mulai memisahkan kedua kelas dengan baik. Pada $x = \pm 1$, peluangnya sekitar 0.23 dan 0.77. Nilai tersebut belum menunjukkan keyakinan penuh, tetapi arah pemisahannya sudah benar. Iterasi IRLS berikutnya membuat kurva semakin curam di sekitar $x = 0$ karena data terpisah sempurna oleh batas tersebut. Dalam keadaan ini, penaksir kemungkinan maksimum tidak memiliki parameter berhingga dan kemiringannya terus membesar. Perilaku tersebut merupakan sifat data, bukan kegagalan algoritma. Dalam praktik, regularisasi pada $\boldsymbol{\theta}$ digunakan untuk mengendalikan pertumbuhan parameter.

Catatan 5. *Metode ini dapat diperluas langsung ke klasifikasi berkelas banyak*

*(*multiclass*) melalui fungsi* softmax *sebagai pengganti sigmoid, dengan struktur Hessian yang analog: blok-blok matriks berbentuk $X^\top DX$ dengan D kini melibatkan matriks kovarians dari distribusi kategorikal, bukan lagi skalar $\pi_i(1-\pi_i)$. Prinsip dasarnya tetap sama seperti yang sudah kita bangun: dekomposisi Hessian menjadi bagian "$J^\top(\cdot)J$" yang mudah dihitung dan bagian sisa. Pada regresi logistik, bagian sisa tersebut lenyap dengan sendirinya karena model logit linear terhadap parameter.*

Bab 11

Eksperimen Empiris: Prediksi Keselamatan Penumpang Titanic

Teori pada Bab 10 akan terasa lebih berarti ketika diterapkan pada data sungguhan. Karena itu, bab ini menerapkan algoritma IRLS (10.8) pada data kompetisi Titanic yang memuat 891 penumpang (Kaggle, 2026). Hasilnya dibandingkan dengan penurunan gradien. Perbedaan laju konvergensi yang dibahas pada Bab 6 kemudian dapat diamati melalui kurva dan angka.

11.1 Penyiapan Data dan Model

Enam fitur dipakai untuk memprediksi keselamatan: kelas tiket (`Pclass`), jenis kelamin (`Sex`, dikodekan 1 =perempuan), usia (`Age`, nilai kosong diisi median), jumlah saudara/pasangan yang ikut (`SibSp`), jumlah orang tua/anak yang ikut (`Parch`), dan tarif tiket (`Fare`). Data dibagi secara terstratifikasi menjadi 713 data latih dan 178 data uji. Seluruh fitur kemudian distandardisasi dengan statistik data latih. Kolom pertama matriks desain X diisi konstanta 1 sebagai intersep.

Setiap iterasi menghitung $\boldsymbol{\pi}_k = \sigma(X\boldsymbol{\theta}_k)$ dan $D_k = \operatorname{diag}(\pi_{k,i}(1-\pi_{k,i}))$. Sistem IRLS diselesaikan melalui QR pada $\sqrt{D_k}X$, tanpa membentuk $X^\top D_k X$ secara eksplisit. Sebagai pembanding, fungsi objektif yang sama dioptimalkan dengan penurunan gradien dan laju belajar $\alpha = 0.05$.

Catatan 6. *Eksperimen referensi dijalankan dengan GNU Octave 8.4. Karena seluruh langkah, data, dan parameter ditulis dalam Langkah Eksperimen, eksperimen yang sama dapat diikuti di MATLAB, Octave, lembar kerja (*spreadsheet*), atau perangkat lunak numerik lain.*

Langkah Eksperimen

Ikuti urutan berikut sebelum melihat grafik dan tabel.

1. Unduh `train.csv` dari kompetisi publik Titanic di `https://www.kaggle.com/competitions/titanic/data`. Ubah `Sex` menjadi 1 untuk perempuan dan 0 untuk laki-laki. Isi `Age` yang kosong dengan median data latih.

2. Pilih enam fitur `Pclass`, `Sex`, `Age`, `SibSp`, `Parch`, dan `Fare`. Acak baris di dalam masing-masing kelas keselamatan dengan *seed* 42. Ambil 80% dari tiap kelas untuk latih dan sisanya untuk uji.

3. Hitung rerata dan simpangan baku hanya dari data latih. Standardisasi data latih dan uji dengan dua nilai tersebut. Tambahkan kolom intersep dan mulai dari $\boldsymbol{\theta}_0 = \mathbf{0}$.

4. Pada setiap iterasi, hitung $\boldsymbol{\pi}_k$ dari (10.1), bentuk $D_k = \operatorname{diag}(\pi_{k,i}(1-\pi_{k,i}))$, lalu selesaikan persamaan IRLS (10.8). Berhenti jika $\|X^\top(\boldsymbol{\pi}_k - \boldsymbol{y})\| < 10^{-8}$.

5. Ulangi masalah yang sama dengan persamaan penurunan gradien (3.16), menggunakan $\alpha = 0.05$ dan paling banyak 3000 iterasi. Catat entropi-silang, norma gradien, dan akurasi uji.

Hasil acuan adalah 7 iterasi dan akurasi uji 76.97% untuk IRLS. Penurunan gradien belum mencapai toleransi pada iterasi ke-3000, tetapi akurasi ujinya juga 76.97%.

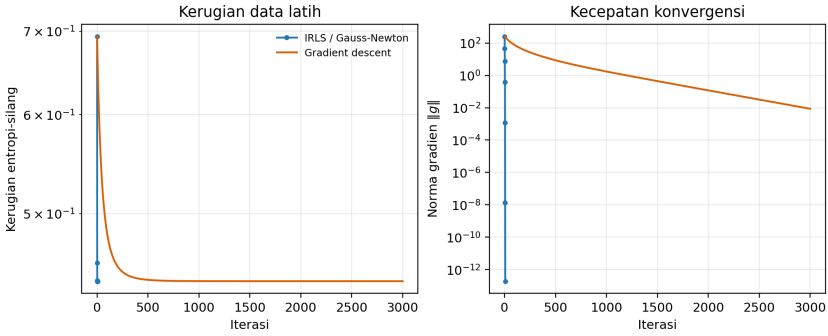
11.2 Hasil: Kecepatan Konvergensi

Tabel 11.1 merangkum jumlah iterasi yang dibutuhkan kedua metode hingga norma gradien $\|X^\top(\boldsymbol{\pi}_k - \boldsymbol{y})\|$ turun di bawah 10^{-8}.

Tabel 11.1: Kecepatan konvergensi pada data Titanic

Metode	Iterasi hingga konvergen	Akurasi data uji
Gauss–Newton (IRLS)	7	**76.97%**
Penurunan gradien ($\alpha = 0.05$)	3000 *(belum konvergen)*	76.97%

Gauss–Newton mencapai toleransi 10^{-8} hanya dalam **7 iterasi**, sedangkan penurunan gradien belum mencapai toleransi yang sama setelah 3000 iterasi. Gambar 11.1 memperlihatkan kerugian entropi-silang dan norma gradien kedua metode. Kedua panel memakai sumbu-y logaritmik agar perbedaan kecepatannya terlihat jelas.



Gambar 11.1: Kurva konvergensi IRLS dan penurunan gradien pada data Titanic

Pola konvergensi yang dibahas pada Bab 6 muncul kembali pada data Titanic. Ketika $\kappa \approx 0$, galat menurun sangat curam. Ketika κ mendekati 1, penurunannya menyerupai garis lurus yang landai pada skala logaritmik. Hasil ini juga konsisten dengan pembahasan pada Subbab 10.3. Karena logit regresi logistik linear terhadap $\boldsymbol{\theta}$, langkah Gauss–Newton berimpit dengan langkah Newton eksak. Akurasi akhir kedua metode pada data Titanic hampir sama; keunggulan utama IRLS terletak pada jumlah iterasi yang jauh lebih sedikit.

11.3 Hasil: Akurasi dan Koefisien Model

Pada data uji, model Gauss–Newton mencapai akurasi 76.97%. Matriks konfusinya disajikan pada Tabel 11.2.

Tabel 11.2: Matriks konfusi klasifikasi Titanic

	Prediksi: tidak selamat	Prediksi: selamat
Aktual: tidak selamat	95	15
Aktual: selamat	26	42

Koefisien model (pada skala fitur yang sudah distandardisasi, sehingga besarnya dapat dibandingkan langsung antarfitur) disajikan pada Tabel 11.3.

Tabel 11.3: Koefisien model logistik pada data Titanic

Fitur	Koefisien $\hat{\theta}$
Intersep	-0.644
Pclass	-0.831
Sex (perempuan= 1)	$+1.348$
Age	-0.447
SibSp	-0.357
Parch	-0.123
Fare	$+0.169$

Tanda koefisien dapat dipakai untuk memeriksa kewajaran model. Koefisien `Sex` paling besar dan positif, sehingga perempuan memiliki peluang selamat yang lebih tinggi pada data ini. Koefisien `Pclass` bertanda negatif; nomor kelas yang lebih besar berkaitan dengan peluang selamat yang lebih rendah. Koefisien `Age` juga negatif, meskipun pengaruhnya lebih kecil. Pola tersebut menunjukkan bahwa optimisasi tidak hanya menurunkan nilai kerugian, tetapi juga menghasilkan hubungan yang dapat ditafsirkan pada data.

Bab 12

Klasifikasi Multikelas dengan Softmax dan Perceptron Satu Lapisan

Bab 10 menunjukkan bagaimana Gauss–Newton dapat digunakan untuk klasifikasi *biner* melalui regresi logistik. Pertanyaan berikutnya muncul secara alami: bagaimana jika jumlah kelas lebih dari dua? Jawabannya membawa kita ke perceptron satu lapisan multikelas. Model ini terdiri atas satu lapisan linear yang diikuti fungsi aktivasi **softmax**. Menariknya, hampir seluruh perangkat yang dibangun sejak Bab 3 masih dapat digunakan.

12.1 Dari Sigmoid ke Softmax

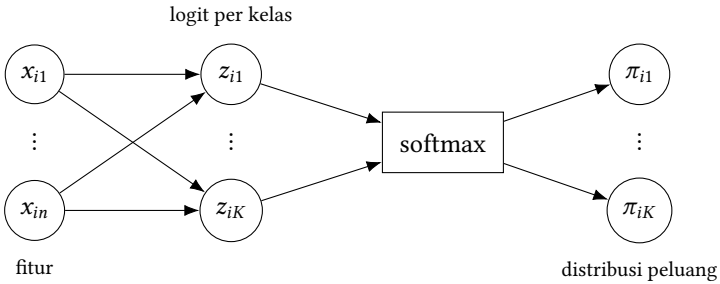
Untuk K kelas, perceptron satu lapisan menghitung logit $z_{ik} = \boldsymbol{x}_i^\top \boldsymbol{\theta}_k$ bagi setiap kelas k. Model ini hanya memiliki satu lapisan bobot dan tidak memakai lapisan tersembunyi. Fungsi **softmax** kemudian mengubah K logit menjadi distribusi peluang:

$$\pi_{ik}(\boldsymbol{\Theta}) = \frac{e^{z_{ik}}}{\sum_{l=1}^{K} e^{z_{il}}}, \qquad k = 1, \dots, K, \qquad \sum_{k=1}^{K} \pi_{ik} = 1. \tag{12.1}$$

Kode MATLAB: Fungsi Softmax

```
clear; clc;
X = [1, -1; 1, 0; 1, 1; 1, 2];
Theta = [0.4, -0.2, 0.0; -0.8, 0.3, 0.5];
Z = X * Theta;
Z = Z - max(Z, [], 2);
expZ = exp(Z);
Pi = expZ ./ sum(expZ, 2);
disp(Pi)
```

Gambar 12.1 memperlihatkan bahwa seluruh logit kelas masuk ke satu blok softmax dan menghasilkan satu distribusi peluang bersama.



Gambar 12.1: Arsitektur perceptron satu lapisan dengan keluaran softmax multikelas

Berbeda dari diagram perceptron satu lapisan biner pada Subbab 10.1, sekarang ada K unit keluaran $(z_{i1}, \dots, z_{iK})$ yang saling terhubung melalui satu blok **softmax** bersama. Inilah yang membuat kelas-kelasnya saling berkaitan, dan yang nanti membuat "bobot" pada Hessian berbentuk matriks blok (12.7), bukan lagi skalar tunggal seperti kasus biner.

Menambahkan konstanta yang sama pada seluruh z_{ik} tidak mengubah peluang softmax. Akibatnya, parameter tidak teridentifikasi secara unik. Masalah ini berasal dari simetri model. Solusinya ialah menetapkan kelas K sebagai **referensi** dengan $\boldsymbol{\theta}_K := \mathbf{0}$. Hanya $\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_{K-1}$ yang menjadi parameter bebas.

Semuanya disusun menjadi vektor

$$\boldsymbol{\theta} = \left(\boldsymbol{\theta}_1;\, \boldsymbol{\theta}_2;\, \ldots;\, \boldsymbol{\theta}_{K-1}\right) \in \mathbb{R}^{n(K-1)}. \tag{12.2}$$

Catatan Aljabar Linear: Softmax dengan $K = 2$ Sama dengan Sigmoid

Sebagai pemeriksaan kewajaran, mari kita pastikan model ini benar-benar memuat Bab 10 sebagai kasus khusus. Untuk $K = 2$ dengan kelas 2 sebagai referensi ($\theta_2 = \mathbf{0}$, sehingga $z_{i2} = 0$), diperoleh

$$\pi_{i1} = \frac{e^{z_{i1}}}{e^{z_{i1}} + e^{z_{i2}}} = \frac{e^{z_{i1}}}{e^{z_{i1}} + 1} = \frac{1}{1 + e^{-z_{i1}}} = \sigma(z_{i1}),$$

yang persis sama dengan fungsi sigmoid (10.1). Jadi, regresi logistik biner bukan model yang terpisah, melainkan kasus $K = 2$ dari kerangka softmax ini.

12.2 Kerugian Entropi-Silang Multikelas dan Turunannya

Label kelas $y_i \in \{1, \ldots, K\}$ kita kodekan sebagai vektor *one-hot* $\mathbf{y}_i \in \{0, 1\}^K$ (semua nol kecuali 1 pada posisi kelas yang benar). Generalisasi (10.2) ke multikelas adalah

$$L(\boldsymbol{\theta}) = -\sum_{i=1}^{m} \sum_{k=1}^{K} y_{ik} \log \pi_{ik}(\boldsymbol{\theta}). \tag{12.3}$$

Setiap logit z_{ik} memengaruhi seluruh peluang $\pi_{i1}, \ldots, \pi_{iK}$ karena semuanya berbagi penyebut normalisasi. Oleh sebab itu, kita memerlukan turunan π_{im} terhadap z_{ik} untuk setiap pasangan kelas m, k. Tulis $Z_i := \sum_l e^{z_{il}}$ sehingga $\pi_{im} = e^{z_{im}}/Z_i$. Aturan hasil bagi memberikan

- Jika $m = k$: $\dfrac{\partial \pi_{ik}}{\partial z_{ik}} = \dfrac{e^{z_{ik}} Z_i - e^{z_{ik}} e^{z_{ik}}}{Z_i^2} = \pi_{ik} - \pi_{ik}^2 = \pi_{ik}(1 - \pi_{ik})$.
- Jika $m \neq k$: $\dfrac{\partial \pi_{im}}{\partial z_{ik}} = \dfrac{0 \cdot Z_i - e^{z_{im}} e^{z_{ik}}}{Z_i^2} = -\pi_{im}\pi_{ik}$.

Kedua kasus ini dapat dirangkum menjadi satu rumus (memakai δ_{mk} dari (12.6), sebab kasus $m = k$ dan $m \neq k$ persis dibedakan oleh δ_{mk}):

$$\frac{\partial \pi_{im}}{\partial z_{ik}} = \pi_{ik}\big(\delta_{mk} - \pi_{im}\big). \tag{12.4}$$

Sekarang kita dapat menurunkan $L_i := -\sum_m y_{im} \log \pi_{im}$ (suku ke-i dari (12.3)) terhadap z_{ik} melalui aturan rantai dengan menjumlahkan pengaruh z_{ik} melalui *setiap* π_{im} yang dipengaruhinya:

$$\begin{aligned}
\frac{\partial L_i}{\partial z_{ik}} &= -\sum_{m=1}^{K} y_{im} \frac{1}{\pi_{im}} \frac{\partial \pi_{im}}{\partial z_{ik}} \\
&= -\sum_{m=1}^{K} \frac{y_{im}}{\pi_{im}} \pi_{ik}\big(\delta_{mk} - \pi_{im}\big) \\
&= -\pi_{ik} \sum_{m=1}^{K} y_{im} \left(\frac{\delta_{mk}}{\pi_{im}} - 1 \right) \\
&= -\pi_{ik} \left(\frac{y_{ik}}{\pi_{ik}} - \sum_{m=1}^{K} y_{im} \right) = \pi_{ik} - y_{ik},
\end{aligned}$$

memakai $\sum_m y_{im}\delta_{mk} = y_{ik}$ (hanya suku $m = k$ yang tersisa dari δ_{mk}) dan $\sum_m y_{im} = 1$ (label *one-hot* selalu menjumlah satu). Hasilnya persis sesederhana kasus biner:

$$\frac{\partial L}{\partial z_{ik}} = \pi_{ik} - y_{ik}. \tag{12.5}$$

Turunan kedua diperoleh dengan menurunkan (12.5) terhadap z_{il}. Karena y_{ik} konstan, hasilnya sama dengan $\partial \pi_{ik}/\partial z_{il}$ yang telah dihitung pada (12.4):

$$\begin{aligned}
\frac{\partial^2 L}{\partial z_{ik} \partial z_{il}} = \frac{\partial \pi_{ik}}{\partial z_{il}} &= \pi_{il}\big(\delta_{kl} - \pi_{ik}\big) \\
&= \pi_{ik}\big(\delta_{kl} - \pi_{il}\big), \\
\delta_{kl} &= \begin{cases} 1, & k = l, \\ 0, & k \neq l. \end{cases}
\end{aligned} \tag{12.6}$$

Kesetaraan dua ekspresi terakhir mudah diperiksa: untuk $k = l$ keduanya sama-sama $\pi_{ik}(1 - \pi_{ik})$, dan untuk $k \neq l$ keduanya sama-sama $-\pi_{ik}\pi_{il}$ (inilah yang menjamin Hessian ini simetris, seperti seharusnya). Inilah yang disebut **Jacobian softmax**. Untuk satu pengamatan i, susun (12.6) sebagai matriks $(K-1) \times (K-1)$:

$$H_i(\boldsymbol{\theta}) = \operatorname{diag}(\boldsymbol{\pi}_i) - \boldsymbol{\pi}_i \boldsymbol{\pi}_i^{\top}, \qquad \boldsymbol{\pi}_i = (\pi_{i1}, \dots, \pi_{i,K-1})^{\top}. \tag{12.7}$$

Contoh Angka

Untuk tiga kelas dengan peluang $(0.5, 0.3, 0.2)$, jika kelas terakhir dijadikan referensi, gunakan dua komponen pertama $\boldsymbol{\pi} = (0.5, 0.3)^{\top}$. Dengan demikian,

$$H_i = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.3 \end{pmatrix} - \begin{pmatrix} 0.5 \\ 0.3 \end{pmatrix} \begin{pmatrix} 0.5 & 0.3 \end{pmatrix} = \begin{pmatrix} 0.25 & -0.15 \\ -0.15 & 0.21 \end{pmatrix}.$$

Elemen diagonal menggambarkan ketidakpastian tiap kelas. Elemen di luar diagonal menunjukkan bahwa perubahan peluang satu kelas memengaruhi kelas lain.

Jika $K = 2$, matriks ini menciut menjadi skalar $\pi_{i1}(1 - \pi_{i1})$, sama dengan turunan kedua pada Subbab 10.2. Dengan demikian, kasus biner merupakan bentuk khusus dari model multikelas.

Catatan Aljabar Linear: Mengapa H_i Selalu Semidefinit Positif

Matriks $H_i = \operatorname{diag}(\boldsymbol{\pi}_i) - \boldsymbol{\pi}_i \boldsymbol{\pi}_i^{\top}$ memiliki interpretasi statistik yang rapi, mirip semangat pembuktian semidefinit-positif pada Bab 3, hanya kali ini melalui argumen variansi, bukan hanya norma. Bayangkan vektor acak *one-hot* $\boldsymbol{e} \in \{0, 1\}^K$ yang bernilai vektor basis $\boldsymbol{e}_k$ dengan peluang π_{ik} (persis distribusi kategorikal yang mendasari label $\boldsymbol{y}_i$). Karena $e_k e_l = e_k$ bila $k = l$ dan 0 bila $k \neq l$, berlaku $\mathbb{E}[\boldsymbol{e}\boldsymbol{e}^{\top}] = \operatorname{diag}(\boldsymbol{\pi}_i)$, sehingga

$$\operatorname{Cov}(\boldsymbol{e}) = \mathbb{E}[\boldsymbol{e}\boldsymbol{e}^{\top}] - \mathbb{E}[\boldsymbol{e}]\mathbb{E}[\boldsymbol{e}]^{\top} = \operatorname{diag}(\boldsymbol{\pi}_i) - \boldsymbol{\pi}_i \boldsymbol{\pi}_i^{\top} = H_i.$$

Catatan Aljabar Linear: Mengapa H_i Selalu Semidefinit Positif *(lanjutan)*

Matriks kovariansi mana pun selalu semidefinit positif. Untuk sembarang $\boldsymbol{a}$, $\boldsymbol{a}^\top H_i \boldsymbol{a} = \mathrm{Var}(\boldsymbol{a}^\top \boldsymbol{e}) \geq 0$, karena variansi tak pernah negatif. Jadi, $H_i \succeq 0$ berlaku otomatis, tanpa syarat tambahan. Hasil ini sama seperti $J^\top J \succeq 0$ pada Bab 3, tetapi kali ini buktinya menggunakan variansi, bukan norma kuadrat. (Keduanya sebenarnya versi berbeda dari argumen yang sama: variansi adalah norma kuadrat pada ruang variabel acak.)

12.3 Sekali Lagi, Gauss–Newton Menjadi Newton Eksak

Seperti pada Subbab 10.3, logit $z_{ik}(\boldsymbol{\theta}) = \boldsymbol{x}_i^\top \boldsymbol{\theta}_k$ linear terhadap parameter. Karena itu, suku yang melibatkan $\nabla^2 z_{ik}$ lenyap untuk setiap $\boldsymbol{\theta}$. Hessian eksak dan aproksimasi Gauss–Newton pun berimpit. Konvergensi lokalnya bersifat kuadratik jika terdapat titik minimum berhingga dan Hessian tetap definit positif di sekitarnya.

12.4 Persamaan Normal Multikelas dan Algoritma IRLS

▲ Pendalaman Teknis

Subbab ini memakai notasi hasil kali Kronecker ($\otimes$) untuk menuliskan persamaan normal multikelas secara ringkas. Notasi ini mungkin terasa asing, tetapi pesan intinya sederhana: bentuknya tetap $J^\top(\cdot) J\, \boldsymbol{p} = -J^\top(\cdot)$ seperti biasa, hanya "bobotnya" berupa matriks blok, bukan lagi skalar. Kita boleh langsung menuju Algoritma IRLS di bawah dan Subbab 12.5 (contoh numerik) tanpa membaca detail notasi Kronecker-nya.

Untuk menuliskan persamaan normalnya secara ringkas, kita pakai notasi hasil kali Kronecker. Misalkan $X \in \mathbb{R}^{m \times n}$ adalah matriks desain yang sama untuk

semua kelas (baris ke-i adalah $\boldsymbol{x}_i^\top$), dan definisikan matriks blok

$$\widetilde{X} = I_{K-1} \otimes X \;\in\; \mathbb{R}^{m(K-1)\times n(K-1)}, \tag{12.8}$$

sehingga $\boldsymbol{z} = \widetilde{X}\boldsymbol{\theta}$, dengan $\boldsymbol{\theta}$ dari (12.2), menumpuk logit seluruh kelas bebas untuk seluruh data. Gradien dan Hessian totalnya menjadi

$$\nabla L(\boldsymbol{\theta}) = \widetilde{X}^\top\big(\boldsymbol{\pi} - \boldsymbol{y}\big), \tag{12.9}$$

$$\nabla^2 L(\boldsymbol{\theta}) = \widetilde{X}^\top \mathcal{D}(\boldsymbol{\theta})\, \widetilde{X}, \tag{12.10}$$

dengan $\boldsymbol{\pi}, \boldsymbol{y} \in \mathbb{R}^{m(K-1)}$ sebagai tumpukan π_{ik} dan y_{ik} untuk $k = 1, \dots, K-1$, serta $\mathcal{D}(\boldsymbol{\theta})$ sebagai matriks blok berukuran $m(K-1) \times m(K-1)$ yang disusun dari blok $(K-1)\times(K-1)$ pada (12.7) untuk tiap i (satu H_i per pengamatan, diselang-seling menurut struktur Kronecker). Dari (12.9)–(12.10), persamaan normalnya adalah

$$\boxed{\widetilde{X}^\top \mathcal{D}(\boldsymbol{\theta}_k)\, \widetilde{X}\; \boldsymbol{p}_k = -\widetilde{X}^\top(\boldsymbol{\pi}_k - \boldsymbol{y})} \tag{12.11}$$

Bandingkan dengan persamaan normal berbobot (8.4) pada Bab 8, dan persamaan IRLS biner (10.8) pada Subbab 10.4: bentuknya tetap $J^\top(\cdot)J\,\boldsymbol{p} = -J^\top(\cdot)$, hanya "bobot" D_k yang skalar sekarang menjadi matriks blok $\mathcal{D}(\boldsymbol{\theta}_k)$ karena kelas-kelas saling berkaitan melalui penyebut bersama pada softmax (12.1).

Algoritma: IRLS untuk Softmax Multikelas

1. Tetapkan kelas referensi K ($\boldsymbol{\theta}_K = \mathbf{0}$). Berikan tebakan awal $\boldsymbol{\theta}_0$ (misalnya $\mathbf{0}$), toleransi $\varepsilon > 0$, $k = 0$.
2. Hitung logit $z_{i,l} = \boldsymbol{x}_i^\top \boldsymbol{\theta}_{k,l}$ untuk $l = 1, \dots, K-1$, lalu $\boldsymbol{\pi}_i$ melalui (12.1).
3. Bentuk blok Hessian per data $H_i = \operatorname{diag}(\boldsymbol{\pi}_i) - \boldsymbol{\pi}_i\boldsymbol{\pi}_i^\top$, susun menjadi $\mathcal{D}(\boldsymbol{\theta}_k)$.
4. Dengan $\widetilde{X}$ pada (12.8), selesaikan $\widetilde{X}^\top \mathcal{D}(\boldsymbol{\theta}_k)\widetilde{X}\,\boldsymbol{p}_k = -\widetilde{X}^\top(\boldsymbol{\pi}_k - \boldsymbol{y})$ (melalui QR/SVD seperti Bab 4 bila hampir singular).

Prosedur (lanjutan)

5. Perbarui $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{p}_k$.
6. Jika $\|\widetilde{X}^\top(\boldsymbol{\pi}_k - \boldsymbol{y})\| < \varepsilon$, berhenti. Jika belum, $k \leftarrow k+1$, kembali ke langkah 2.

12.5 Contoh Numerik: Tiga Kelas

Tinjau $K = 3$ kelas (label 0, 1, 2; kelas 2 sebagai referensi) dengan satu fitur x (plus intersep, $n = 2$), dan $m = 6$ data yang tersusun rapi pada Tabel 12.1. Terdapat

Tabel 12.1: Contoh data klasifikasi multikelas

i	1	2	3	4	5	6
x_i	-2	-1	0	1	2	3
y_i	0	0	1	1	2	2

empat parameter bebas, yaitu $\boldsymbol{\theta} = (\theta_{1,\text{intersep}}, \theta_{1,\text{kemiringan}}, \theta_{2,\text{intersep}}, \theta_{2,\text{kemiringan}})$, untuk kelas 0 dan 1 (kelas 2 tetap $\boldsymbol{\theta}_2 = \mathbf{0}$). Pada tebakan awal $\boldsymbol{\theta}_0 = \mathbf{0}$, seluruh logit nol, sehingga $\pi_{i0} = \pi_{i1} = \pi_{i2} = \tfrac{1}{3}$ untuk keenam data. Situasi seragam ini mirip dengan contoh pada Subbab 10.5.

Karena $\pi_{ik} = \tfrac{1}{3}$ seragam, blok Hessian per data juga seragam: $H_i = \operatorname{diag}(\tfrac{1}{3}, \tfrac{1}{3}) - \tfrac{1}{3}\tfrac{1}{3}\mathbf{1}\mathbf{1}^\top = \begin{pmatrix} 2/9 & -1/9 \\ -1/9 & 2/9 \end{pmatrix}$ untuk setiap i (lihat (12.7)). Setelah dijumlahkan melalui $\widetilde{X}^\top D \widetilde{X}$ atas keenam data (dengan $\sum x_i = 3$, $\sum x_i^2 = 19$), diperoleh Hessian 4×4

$$\nabla^2 L(\boldsymbol{\theta}_0) = \begin{pmatrix} 1.333 & 0.667 & -0.667 & -0.333 \\ 0.667 & 4.222 & -0.333 & -2.111 \\ -0.667 & -0.333 & 1.333 & 0.667 \\ -0.333 & -2.111 & 0.667 & 4.222 \end{pmatrix}, \tag{12.12}$$

dan gradiennya $\nabla L(\boldsymbol{\theta}_0) = (0,\ 4,\ 0,\ 0)^\top$. Hanya komponen kemiringan kelas 0 yang tidak nol. Komponen intersep bernilai nol karena setiap kelas memiliki dua

anggota, sehingga jumlah label sama dengan jumlah peluang awal. Komponen kemiringan kelas 1 juga nol karena jumlah x_i pada kelas tersebut sama dengan sepertiga dari $\sum x_i$. Menyelesaikan $\nabla^2 L(\boldsymbol{\theta}_0)\, \boldsymbol{p}_0 = -\nabla L(\boldsymbol{\theta}_0)$ dengan Hessian (12.12) memberi

$$\boldsymbol{p}_0 \approx (0.686,\ -1.371,\ 0.343,\ -0.686), \qquad \boldsymbol{\theta}_1 = \boldsymbol{\theta}_0 + \boldsymbol{p}_0. \tag{12.13}$$

Setelah satu iterasi ini, ketiga kurva peluang $\pi_0(x), \pi_1(x), \pi_2(x) = 1 - \pi_0(x) - \pi_1(x)$ sudah mulai memisahkan tiga kelompok data.

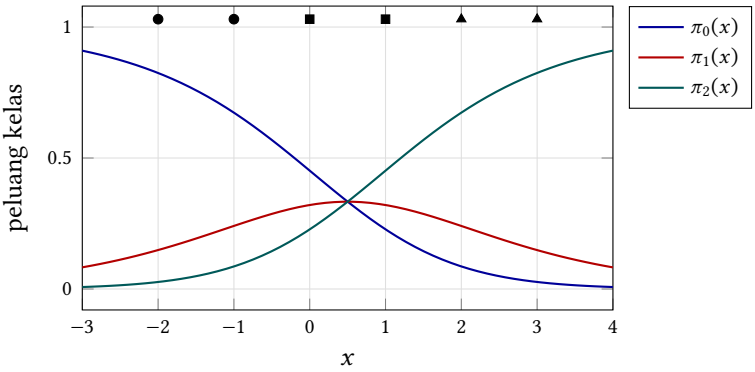
Langkah Eksperimen

Sebelum membuat grafik, lakukan urutan berikut.

1. Masukkan data $(-2, 0), (-1, 0), (0, 1), (1, 1), (2, 2), (3, 2)$. Pilih kelas 2 sebagai referensi.
2. Bentuk $\boldsymbol{x}_i = (1, x_i)$ dan mulai dari seluruh parameter bernilai nol.
3. Hitung peluang softmax melalui (12.1), lalu bentuk $H_i = \operatorname{diag}(\boldsymbol{\pi}_i) - \boldsymbol{\pi}_i \boldsymbol{\pi}_i^\top$.
4. Susun dan selesaikan persamaan normal multikelas (12.11).
5. Periksa bahwa $\boldsymbol{p}_0 \approx (0.686, -1.371, 0.343, -0.686)$ seperti pada (12.13). Masukkan parameter ini kembali ke fungsi softmax untuk menggambar tiga kurva peluang.

Peluang ketiga kelas setelah satu langkah IRLS ditampilkan pada Gambar 12.2.

Titik-titik di bagian atas grafik (lingkaran, kotak, segitiga) menandai data asli kelas 0, 1, dan 2. Terlihat $\pi_0(x)$ (biru) unggul di sisi kiri, $\pi_2(x)$ (hijau) unggul di sisi kanan, dan $\pi_1(x)$ (merah) hanya sempat unggul tipis di tengah. Hasil ini masuk akal karena dengan hanya satu iterasi dan satu fitur, model belum sepenuhnya yakin, tetapi arah pemisahan tiga kelasnya sudah benar. Iterasi berikutnya, mengikuti Algoritma IRLS di atas, akan mempertajam ketiga kurva ini lebih lanjut.



Gambar 12.2: Kurva peluang tiga kelas setelah satu iterasi IRLS pada contoh softmax

Catatan 7. *Struktur di atas menjadi dasar GGN pada lapisan keluaran softmax. Pada jaringan berlapis, $z_{ik}(\boldsymbol{\theta})$ tidak lagi linear terhadap seluruh parameter. Karena itu, suku $M(\boldsymbol{\theta})$ tidak lenyap secara umum dan GGN kembali menjadi aproksimasi, bukan Hessian eksak.*

12.6 Eksperimen Publik: Iris dan Softmax

Contoh tiga kelas sebelumnya sengaja berukuran kecil agar persamaan dapat dihitung dengan tangan. Sekarang algoritma yang sama diterapkan pada data publik. Dataset Iris berisi 150 bunga, empat fitur numerik, dan tiga kelas. Data dapat diunduh dari UCI Machine Learning Repository (Fisher, 1936).

Langkah Eksperimen

Ikuti urutan berikut agar hasilnya dapat dibandingkan dengan eksperimen buku.

1. Unduh data Iris. Gunakan empat kolom pertama sebagai fitur dan kolom terakhir sebagai kelas $0, 1, 2$. Tidak diperlukan kode program khusus; lembar kerja (*spreadsheet*) sudah cukup untuk memeriksa ukuran data dan membuat tabel.

Langkah Eksperimen *(lanjutan)*

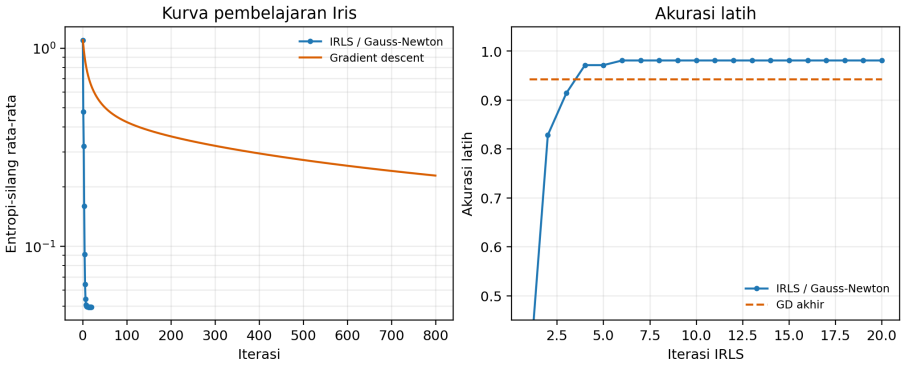
2. Dengan *seed* acak 42, acak indeks *di dalam tiap kelas*. Pada eksperimen acuan, satu generator `numpy.random.default_rng(42)` dipakai berurutan untuk kelas 0, 1, dan 2. Ambil 35 bunga per kelas untuk data latih dan 15 bunga per kelas untuk data uji. Hasilnya adalah 105 data latih dan 45 data uji.

3. Hitung rerata dan simpangan baku setiap fitur hanya dari 105 data latih. Standardisasi data latih dan uji dengan dua nilai tersebut, lalu tambahkan kolom intersep 1.

4. Tetapkan kelas 2 sebagai kelas referensi dan mulai dari semua parameter nol. Pada setiap iterasi IRLS, hitung softmax, bentuk blok $H_i = \operatorname{diag}(\boldsymbol{\pi}_i) - \boldsymbol{\pi}_i \boldsymbol{\pi}_i^\top$, susun matriks blok, selesaikan persamaan normal, dan perbarui parameter. Jalankan 20 iterasi.

5. Sebagai pembanding, gunakan parameter awal yang sama dan pembaruan penurunan gradien dengan $\alpha = 0.1$ selama 800 iterasi. Catat entropi-silang data latih dan akurasi data uji.

Gambar 12.3 menunjukkan kurva pembelajaran dan akurasi pada data Iris.



Gambar 12.3: Kurva pembelajaran dan akurasi latih pada data Iris

Hasil eksperimen ditunjukkan pada Tabel 12.2. Pada data Iris, IRLS mencapai

akurasi uji sempurna pada pembagian ini. Penurunan gradien juga belajar dengan baik, tetapi satu bunga dari kelas 2 masih diprediksi sebagai kelas 1.

Tabel 12.2: Hasil klasifikasi multikelas pada data Iris

Metode	Iterasi	Entropi-silang akhir	Akurasi uji
IRLS / Gauss–Newton	20	0.0495	100.00%
penurunan gradien ($\alpha = 0.1$)	800	0.2275	97.78%

Untuk IRLS, matriks konfusi pada 45 data uji disajikan pada Tabel 12.3.

Tabel 12.3: Matriks konfusi IRLS pada data Iris

	Prediksi 0	Prediksi 1	Prediksi 2
Aktual 0	15	0	0
Aktual 1	0	15	0
Aktual 2	0	0	15

Eksperimen ini memperlihatkan alasan contoh sintetis tetap berguna. Data publik memberi konteks dan ukuran performa yang nyata, sedangkan contoh enam titik tetap menjadi tempat terbaik untuk memahami setiap blok Hessian secara manual. Keduanya saling melengkapi: yang satu menjelaskan mekanisme, sedangkan yang lain menguji apakah mekanisme tersebut bekerja pada data yang lebih realistis.

Bab 13

Jaringan Berlapis: MLP dengan Satu Lapisan Tersembunyi

Pada perceptron satu lapisan, logit linear terhadap parameter sehingga Gauss–Newton berimpit dengan Newton eksak. Keistimewaan itu hilang ketika sebuah lapisan tersembunyi dengan aktivasi nonlinear ditambahkan. Bab ini menelusuri perubahan tersebut. Hasilnya akan menjelaskan mengapa redaman Levenberg–Marquardt menjadi kebutuhan praktis pada jaringan berlapis (Hagan dan Menhaj, 1994).

13.1 Model MLP sebagai Wujud Konkret dari $f(\boldsymbol{x}, \boldsymbol{\theta})$ pada Bab 1

Ingat kembali rumusan paling awal buku ini: kita ingin mencocokkan model umum $f(\boldsymbol{x}, \boldsymbol{\theta})$ dengan data (x_i, y_i) melalui residu $r_i(\boldsymbol{\theta}) = y_i - f(x_i, \boldsymbol{\theta})$. Sejak Bab 2 hingga 6, f boleh berbentuk apa saja. Kita tidak pernah membutuhkan bentuk eksplisitnya, hanya turunannya. *Multilayer perceptron* (MLP) dengan satu lapisan tersembunyi adalah salah satu wujud paling kaya dari f semacam itu: alih-alih satu lapisan bobot seperti pada Bab 10 dan 12, sekarang ada dua

lapisan yang disusun berurutan. Untuk fitur skalar x dan H unit tersembunyi,

$$a_j = w_j^{(1)} x + b_j^{(1)}, \qquad h_j = \varphi(a_j), \qquad j = 1, \dots, H, \tag{13.1}$$

$$f(x, \boldsymbol{\theta}) = \sum_{j=1}^{H} w_j^{(2)} h_j + b^{(2)}, \tag{13.2}$$

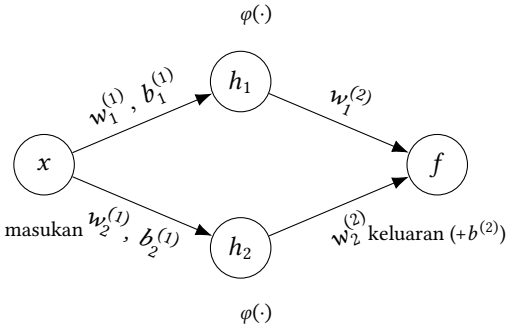
$$\boldsymbol{\theta} = \left(w_1^{(1)}, \dots, w_H^{(1)}, b_1^{(1)}, \dots, b_H^{(1)}, w_1^{(2)}, \dots, w_H^{(2)}, \, b^{(2)}\right),$$

dengan φ sebagai fungsi aktivasi (misalnya tanh) yang membuat lapisan tersembunyi bukan sekadar transformasi linear lagi. Berikut ini adalah sketsanya untuk $H = 2$, yang akan menjadi contoh kita di sepanjang bagian ini:

Kode MATLAB: Propagasi Maju MLP Satu Lapisan Tersembunyi

```
clear; clc;
x = linspace(-1, 1, 5)';
w1 = [0.7; -0.4];
b1 = [0.1; 0.2];
w2 = [0.8; -0.6];
b2 = 0.05;
a = w1*x' + b1;
h = tanh(a);
f = w2'*h + b2;
disp(f)
```

Arsitektur untuk $H = 2$ ditunjukkan pada Gambar 13.1.



Gambar 13.1: MLP dengan satu lapisan tersembunyi dan dua unit tersembunyi

Vektor $\boldsymbol{\theta}$ sekarang berdimensi $3H + 1$; untuk $H = 2$, model memiliki tujuh parameter. Hal yang lebih penting adalah bahwa $f(x, \boldsymbol{\theta})$ tidak lagi linear terhadap $\boldsymbol{\theta}$. Parameter $w_j^{(1)}$ dan $b_j^{(1)}$ memengaruhi keluaran melalui komposisi fungsi aktivasi $\varphi(\cdot)$ yang melengkung.

13.2 Propagasi Balik sebagai Perhitungan Matriks Jacobian

Untuk memakai Gauss–Newton, kita memerlukan $J(\boldsymbol{\theta})$ yang berisi $\partial f_i / \partial \theta_l$. Karena f tersusun dari dua lapisan, turunannya membutuhkan aturan rantai dua tingkat. Penyusunan aturan rantai secara sistematis inilah yang disebut **propagasi balik** (Rumelhart dkk., 1986).

Untuk parameter lapisan luar $w_j^{(2)}$ dan $b^{(2)}$, turunannya dapat langsung dihitung karena $f = \sum_{j'} w_{j'}^{(2)} h_{j'} + b^{(2)}$ adalah fungsi *linear* terhadap keduanya (persis seperti menurunkan $ax + b$ terhadap a atau b):

$$\frac{\partial f}{\partial w_j^{(2)}} = h_j, \qquad \frac{\partial f}{\partial b^{(2)}} = 1. \tag{13.3}$$

Untuk parameter lapisan dalam $w_j^{(1)}$ dan $b_j^{(1)}$, keadaannya berbeda: keduanya memengaruhi f secara *tidak langsung*, melalui rantai $w_j^{(1)} \to a_j \to h_j \to f$. Aturan rantai untuk rantai tiga langkah ini adalah perkalian tiga turunan, masing-masing dihitung satu langkah:

$$\frac{\partial f}{\partial w_j^{(1)}} = \underbrace{\frac{\partial f}{\partial h_j}}_{\text{langkah 1}} \cdot \underbrace{\frac{\partial h_j}{\partial a_j}}_{\text{langkah 2}} \cdot \underbrace{\frac{\partial a_j}{\partial w_j^{(1)}}}_{\text{langkah 3}}. \tag{13.4}$$

Ketiga faktor pada (13.4) dihitung secara berurutan. Pertama, sifat linear $f = \sum_{j'} w_{j'}^{(2)} h_{j'} + b^{(2)}$ memberikan $\partial f / \partial h_j = w_j^{(2)}$; suku dengan $j' \neq j$ tidak bergantung pada h_j sehingga turunannya nol. Kedua, definisi turunan fungsi aktivasi memberikan $\partial h_j / \partial a_j = \varphi'(a_j)$. Ketiga, karena $a_j = w_j^{(1)} x + b_j^{(1)}$

linear terhadap $w_j^{(1)}$, berlaku $\partial a_j / \partial w_j^{(1)} = x$. Perkalian ketiga hasil tersebut memberikan

$$\frac{\partial f}{\partial w_j^{(1)}} = w_j^{(2)}\, \varphi'(a_j)\, x, \qquad \frac{\partial f}{\partial b_j^{(1)}} = w_j^{(2)}\, \varphi'(a_j), \tag{13.5}$$

Rumus kedua diperoleh dengan cara yang sama. Hanya langkah ketiga yang berubah menjadi $\partial a_j / \partial b_j^{(1)} = 1$, karena $a_j = w_j^{(1)} x + b_j^{(1)}$ linear terhadap $b_j^{(1)}$. Pola "kalikan mundur dari keluaran ke masukan" ini adalah asal-usul nama propagasi balik. Sinyal $w_j^{(2)}$ dikalikan dengan kepekaan lokal $\varphi'(a_j)$, lalu dikalikan dengan masukan x atau 1.

Contoh Angka

Ambil satu unit tersembunyi dengan aktivasi tanh. Misalkan $x = 2$, $w^{(1)} = 0.5$, $b^{(1)} = 0$, $w^{(2)} = 3$, dan $b^{(2)} = 1$. Ikuti rantainya.

1. Maju ke unit tersembunyi: $a = 0.5(2) + 0 = 1$.
2. Aktifkan: $h = \tanh(1) \approx 0.762$.
3. Hitung keluaran: $f = 3(0.762) + 1 \approx 3.286$.
4. Hitung kepekaan lokal: $\varphi'(a) = 1 - h^2 \approx 0.420$.
5. Turunkan mundur: $\partial f / \partial w^{(1)} \approx 3(0.420)(2) = 2.520$, $\partial f / \partial b^{(1)} \approx 1.260$, $\partial f / \partial w^{(2)} \approx 0.762$, dan $\partial f / \partial b^{(2)} = 1$.

Jika yang dipakai adalah residu $r = y - f$, semua turunan di atas diberi tanda minus ketika disusun menjadi satu baris Jacobian.

Kode MATLAB: Mengisi Satu Baris J melalui Propagasi Balik

```
clear; clc;
x = 0.5;
w1 = [0.7; -0.4];
b1 = [0.1; 0.2];
w2 = [0.8; -0.6];
b2 = 0.05;
```

Kode MATLAB: Mengisi Satu Baris J melalui Propagasi Balik *(lanjutan)*

```
a = w1*x + b1; h = tanh(a);
dphi = 1 - h.^2;
df_dw2 = h;
df_db2 = 1;
df_dw1 = w2 .* dphi .* x;
df_db1 = w2 .* dphi;
J_row = -[df_dw1; df_db1; df_dw2; df_db2]';
disp(J_row)
```

Catatan Aljabar Linear: Propagasi Balik Mengisi Baris-Baris J

Persamaan (13.3)–(13.5) mengisi satu baris Jacobian untuk satu data x_i. Kolom-kolomnya mengikuti urutan parameter $w_1^{(1)}, \dots, b^{(2)}$. Setelah proses yang sama dilakukan pada seluruh m data, terbentuk $J \in \mathbb{R}^{m\times(3H+1)}$. Jadi, propagasi balik tidak hanya menghitung gradien. Prosedur ini menghitung Jacobian, sedangkan $\nabla S = J^\top \boldsymbol{r}$ merupakan hasil lanjutannya.

13.3 Mengapa Suku $M(\theta)$ Kini Tidak Lagi Nol

Ingat dekomposisi Hessian eksak pada (3.5): $H(\boldsymbol{\theta}) = J(\boldsymbol{\theta})^\top J(\boldsymbol{\theta}) + M(\boldsymbol{\theta})$, dengan $M(\boldsymbol{\theta}) = \sum_i r_i(\boldsymbol{\theta}) \nabla^2 r_i(\boldsymbol{\theta})$. Pada Bab 10 dan 12, $M(\boldsymbol{\theta}) \equiv \mathbf{0}$ untuk semua $\boldsymbol{\theta}$ karena logitnya linear terhadap parameter. Sekarang mari kita periksa: apakah itu masih berlaku untuk MLP?

Sebagai contoh, ambil turunan kedua terhadap $w_j^{(1)}$ sebanyak dua kali:

$$\frac{\partial^2 f}{\partial (w_j^{(1)})^2} = w_j^{(2)}\, \varphi''(a_j)\, x^2. \tag{13.6}$$

Berbeda dari (13.5), suku ini melibatkan turunan kedua fungsi aktivasi. Untuk $\varphi = \tanh$, nilai $\varphi''(a) = -2\tanh(a)(1 - \tanh^2(a))$ tidak nol pada hampir semua a. Karena $r_i = y_i - f_i$, turunan kedua ini menyumbang pada $M(\boldsymbol{\theta})$. Karena itu, dengan aktivasi yang melengkung, $M(\boldsymbol{\theta})$ secara umum tidak nol. Matriks $J^\top J$ kembali menjadi aproksimasi Hessian, bukan Hessian eksak seperti pada Bab 10 dan 12.

Catatan 8. *Ada pengecualian menarik: jika φ adalah fungsi aktivasi* piecewise-linear *seperti ReLU ($\varphi(a) = \max(0, a)$), maka $\varphi''(a) = 0$ di hampir setiap titik (kecuali tepat di $a = 0$, tempat turunannya tidak terdefinisi). Akibatnya, suku* (13.6) *lenyap hampir di mana-mana, dan $M(\boldsymbol{\theta}) \approx \mathbf{0}$ kembali. Inilah salah satu alasan (di luar alasan komputasi) mengapa aktivasi ReLU disukai dalam praktik pembelajaran mendalam: aktivasi ini membuat aproksimasi Gauss–Newton lebih dekat ke Hessian eksak dibandingkan aktivasi mulus seperti* tanh *atau sigmoid.*

13.4 *Generalized Gauss–Newton* dan Perlunya Regularisasi

Meskipun $M(\boldsymbol{\theta}) \neq \mathbf{0}$, aproksimasi $H_{\text{GGN}}(\boldsymbol{\theta}) := J(\boldsymbol{\theta})^\top J(\boldsymbol{\theta})$ tetap layak digunakan (Schraudolph, 2002). Matriks ini selalu semidefinit positif untuk setiap J, sebagaimana dibuktikan pada Bab 3. Pembuktian tersebut tidak mensyaratkan Jacobian konstan atau model linear. Sifat ini penting karena Hessian eksak pada permukaan kerugian MLP dapat memiliki arah kelengkungan negatif. Dalam keadaan itu, arah Newton $-H^{-1}\nabla S$ belum tentu merupakan arah turun. Aproksimasi GGN menghindari masalah tersebut karena kepositifannya dijamin oleh konstruksi, sesuai hasil pada Bab 5.

Konsekuensinya bersifat praktis. Jacobian kini berubah secara nonlinear terhadap parameter, sehingga aproksimasi (3.9) hanya akurat untuk langkah yang cukup kecil. Langkah penuh dapat melenceng jauh. Levenberg–Marquardt membatasi langkah melalui radius kepercayaan, sehingga iterasi tetap berada di wilayah tempat model lokal layak dipercaya. Contoh berikut memperlihatkan perbedaannya secara numerik.

Algoritma: Gauss–Newton/Levenberg–Marquardt untuk MLP

1. Berikan tebakan awal $\boldsymbol{\theta}_0$, $\lambda_0 > 0$, toleransi $\varepsilon > 0$, $k = 0$.

2. Lakukan propagasi maju. Hitung a_j, h_j, f_i untuk seluruh data melalui (13.1)–(13.2), kemudian bentuk $\boldsymbol{r}_k$.

Prosedur (lanjutan)

3. Lakukan propagasi balik untuk menghitung J_k melalui (13.3)–(13.5).
4. Selesaikan $\left(J_k^\top J_k + \lambda_k I\right)\boldsymbol{p}_k = -J_k^\top \boldsymbol{r}_k$ (Bab 7).
5. Jika $S(\boldsymbol{\theta}_k + \boldsymbol{p}_k) < S(\boldsymbol{\theta}_k)$: terima $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{p}_k$, kecilkan λ_{k+1}. Jika tidak: tolak langkah, besarkan λ_k, ulangi langkah 4.
6. Jika $\|J_k^\top \boldsymbol{r}_k\| < \varepsilon$, berhenti. Jika belum, $k \leftarrow k+1$, kembali ke langkah 2.

13.5 Contoh Numerik: Mengapa Langkah Mentah Dapat Gagal

Tinjau MLP dengan $H = 2$ unit tersembunyi beraktivasi tanh, dilatih untuk mencocokkan $y = \sin(x)$ pada $m = 8$ titik $x = -3, -2, -1, -0.5, 0.5, 1, 2, 3$. Tebakan awal $\boldsymbol{\theta}_0$ diberikan oleh

$$\left(w_1^{(1)}, b_1^{(1)}, w_2^{(1)}, b_2^{(1)}, w_1^{(2)}, w_2^{(2)}, b^{(2)}\right) = (0.7,\ 0.1,\ -0.4,\ 0.2,\ 0.8,\ -0.6,\ 0.05).$$

Nilai objektif awalnya adalah $S(\boldsymbol{\theta}_0) = 1.339$. Matriks Jacobian $J \in \mathbb{R}^{8\times 7}$ dihitung melalui (13.3)–(13.5) untuk kedelapan titik data (baris pertama, sebagai contoh, adalah $(0.170, -0.057, -0.389, 0.130, 0.964, -0.885, -1.000)$ untuk $x_1 = -3$), dan berperingkat penuh kolom ($\operatorname{rank}(J) = 7$).

Hal yang menarik justru terlihat saat kita membandingkan norma kedua suku dekomposisi Hessian: dalam norma Frobenius, $\|J^\top J\|_F \approx 14.57$ sedangkan $\|M(\boldsymbol{\theta}_0)\|_F \approx 7.90$. Suku yang "dibuang" oleh Gauss–Newton besarnya lebih dari separuh suku yang dipertahankan! Keadaan ini jelas berbeda jauh dari situasi $M \equiv \mathbf{0}$ pada Bab 10–12.

Langkah Eksperimen

Sebelum membaca tabel, lakukan perhitungan berikut.

1. Gunakan $x = (-3, -2, -1, -0.5, 0.5, 1, 2, 3)$ dan tetapkan $y_i = \sin(x_i)$.
2. Hitung h_j dan f melalui (13.1)–(13.2) dengan parameter awal $(0.7, 0.1, -0.4, 0.2, 0.8, -0.6, 0.05)$.
3. Hitung setiap baris Jacobian melalui (13.3)–(13.5), lalu susun J.
4. Hitung $S = \frac{1}{2}\|\boldsymbol{r}\|^2$ dan selesaikan (3.13) untuk langkah Gauss–Newton tanpa redaman.
5. Ulangi penyelesaian dengan (7.1) dan $\lambda = 1$.
6. Periksa $S_0 \approx 1.339$, $S_{\mathrm{GN}} \approx 802.9$, dan $S_{\mathrm{LM}} \approx 0.538$ sebelum membaca interpretasi hasil.

Dampaknya terlihat nyata ketika kita mencoba langkah Gauss–Newton murni ($\alpha_k = 1$, tanpa redaman). Hasilnya dirangkum pada Tabel 13.1. Langkah

Tabel 13.1: Dampak redaman dan lintasan LM pada MLP satu lapisan tersembunyi

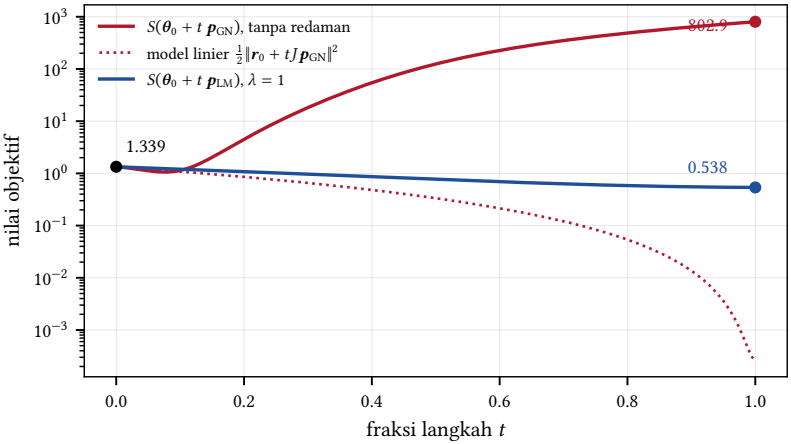
Langkah	$S(\boldsymbol{\theta}_0 + \boldsymbol{p})$	
Sebelum melangkah	1.339	
Gauss–Newton murni ($\lambda = 0$)	802.900	*(melenceng jauh!)*
Levenberg–Marquardt ($\lambda = 1$)	0.538	*(menurun sehat)*

Iterasi k	0	1	2	3	4	5	6
$S(\boldsymbol{\theta}_k)$	1.339	0.538	0.404	0.303	0.276	0.215	0.215

Gauss–Newton tanpa redaman menaikkan S dari 1.339 menjadi 802.9. Sebaliknya, Levenberg–Marquardt menurunkannya menjadi 0.538 dan mempertahankan penurunan pada langkah berikutnya. Hasil ini memperlihatkan pentingnya redaman pada optimisasi MLP nonlinear.

Gambar 13.2 menjelaskan apa yang terjadi di antara titik awal dan ujung

langkah. Arah Gauss–Newton tetap merupakan arah turun, sesuai Proposisi 1. Nilai S memang turun sedikit hingga sekitar 1.06 pada fraksi langkah $t \approx 0.075$, lalu naik tajam sampai 802.9 pada $t = 1$. Model linear (3.9) justru meramalkan S hampir nol pada $t = 1$. Selisih inilah yang menunjukkan bahwa model lokal hanya dapat dipercaya untuk langkah pendek. Arah Levenberg–Marquardt lebih pendek dan menurunkan S secara stabil sepanjang lintasannya.



Gambar 13.2: Nilai objektif sepanjang arah Gauss–Newton dan arah Levenberg–Marquardt ($\lambda = 1$) pada MLP satu lapisan tersembunyi, beserta ramalan model linear untuk arah Gauss–Newton

Bab 14

MLP dengan Dua Lapisan Tersembunyi

Bab 13 menurunkan Jacobian MLP dengan satu lapisan tersembunyi melalui rumus eksplisit. Rumus tersebut masih dapat ditulis dengan ringkas. Ketika lapisan tersembunyi kedua ditambahkan, cara ini tidak lagi praktis. Di sinilah struktur propagasi balik sebagai **rekursi mundur** mulai terlihat. Bab ini menurunkan rekursi tersebut dan menunjukkan fenomena numerik penting pada jaringan yang lebih dalam. Matriks Jacobian dapat berperingkat penuh secara teoretis, tetapi tetap tidak stabil untuk digunakan secara numerik.

14.1 Model MLP Dua Lapisan Tersembunyi

Perluas model (13.1)–(13.2) dengan menyisipkan satu lapisan tersembunyi lagi di antara $h^{(1)}$ dan keluaran. Untuk fitur skalar x, dengan H_1 unit pada lapisan tersembunyi pertama dan H_2 unit pada lapisan tersembunyi kedua:

$$a_j^{(1)} = w_j^{(1)} x + b_j^{(1)}, \qquad h_j^{(1)} = \varphi(a_j^{(1)}), \qquad j = 1, \dots, H_1, \tag{14.1}$$

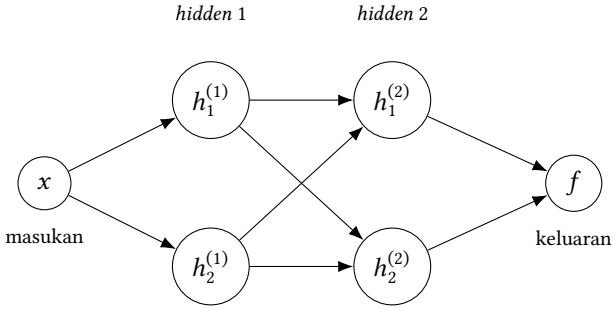
$$a_k^{(2)} = \sum_{j=1}^{H_1} W_{kj}^{(2)} h_j^{(1)} + b_k^{(2)}, \qquad h_k^{(2)} = \varphi(a_k^{(2)}), \qquad k = 1, \dots, H_2, \tag{14.2}$$

$$f(x, \boldsymbol{\theta}) = \sum_{k=1}^{H_2} w_k^{(3)} h_k^{(2)} + b^{(3)}. \tag{14.3}$$

Kode MATLAB: Propagasi Maju MLP Dua Lapisan Tersembunyi

```
clear; clc;
x = 0.5;
w1 = [0.7; -0.4];
b1 = [0.1; 0.2];
W2 = [0.5, -0.3; 0.2, 0.6];
b2 = [0.1; -0.1];
w3 = [0.8; -0.6];
b3 = 0.05;
a1 = w1*x + b1; h1 = tanh(a1);
a2 = W2*h1 + b2; h2 = tanh(a2);
f  = w3'*h2 + b3;
disp(f)
```

Perbedaannya dengan model pada Bab 13 tampak jelas pada (14.2). Unit-unit lapisan tersembunyi kedua ($a_k^{(2)}$) menerima masukan dari *seluruh* unit lapisan tersembunyi pertama melalui matriks bobot $W^{(2)} \in \mathbb{R}^{H_2 \times H_1}$, bukan langsung dari x. Arsitektur untuk $H_1 = H_2 = 2$ diperlihatkan pada Gambar 14.1.



Gambar 14.1: MLP dengan dua lapisan tersembunyi dan dua unit pada setiap lapisan

Total parameter $\boldsymbol{\theta} = (w_j^{(1)}, b_j^{(1)}, W_{kj}^{(2)}, b_k^{(2)}, w_k^{(3)}, b^{(3)})$ berdimensi $2H_1 + H_1H_2 + 2H_2 + 1$. Untuk $H_1 = H_2 = 2$, model ini memiliki tiga belas parameter, hampir dua kali lipat model pada Bab 13.

14.2 Propagasi Balik sebagai Rekursi Sinyal Galat

▲ Pendalaman Teknis

Subbab ini menurunkan rekursi propagasi balik secara formal melalui aturan rantai dua tingkat. Jika kita hanya ingin mengetahui *hasil akhirnya* tanpa menelusuri seluruh turunannya, kita dapat membaca rumus (14.4)–(14.6) dan kotak Catatan Aljabar Linear setelahnya, lalu melanjutkan ke Subbab 14.3.

Penurunan langsung terhadap setiap parameter menghasilkan jumlah bersarang yang panjang. Propagasi balik merapikan perhitungan itu dengan mendefinisikan sinyal galat

$$\delta_j^{(1)} := \frac{\partial f}{\partial a_j^{(1)}}, \qquad \delta_k^{(2)} := \frac{\partial f}{\partial a_k^{(2)}},$$

kemudian menghitung keduanya secara rekursif dari lapisan keluaran menuju lapisan masukan.

Sinyal $\delta_k^{(2)}$ dihitung persis seperti $\partial f / \partial b_j^{(1)}$ pada Subbab 13.2 (rantai satu jalur $a_k^{(2)} \to h_k^{(2)} \to f$, memakai $f = \sum_k w_k^{(3)} h_k^{(2)} + b^{(3)}$ dan $h_k^{(2)} = \varphi(a_k^{(2)})$):

$$\delta_k^{(2)} = \frac{\partial f}{\partial h_k^{(2)}} \cdot \frac{\partial h_k^{(2)}}{\partial a_k^{(2)}} = w_k^{(3)} \, \varphi'(a_k^{(2)}). \tag{14.4}$$

Sinyal $\delta_j^{(1)}$ lebih rumit karena $h_j^{(1)}$ memengaruhi seluruh unit pada lapisan kedua. Aturan rantai mengharuskan kontribusi dari semua jalur dijumlahkan. Jalur ke-k menyumbang $\delta_k^{(2)} W_{kj}^{(2)}$. Setelah seluruh jalur dijumlahkan, hasilnya dikalikan dengan $\varphi'(a_j^{(1)})$:

$$\delta_j^{(1)} = \underbrace{\left(\sum_{k=1}^{H_2} \delta_k^{(2)} \, W_{kj}^{(2)} \right)}_{\text{jumlah atas seluruh jalur}} \cdot \underbrace{\varphi'(a_j^{(1)})}_{\text{faktor bersama}},$$

$$= \varphi'(a_j^{(1)}) \sum_{k=1}^{H_2} W_{kj}^{(2)}\, \delta_k^{(2)}. \tag{14.5}$$

Sinyal galat pada lapisan pertama, singkatnya, diperoleh dengan menyebarkan $\delta^{(2)}$ mundur melalui $W^{(2)}$ (dijumlahkan atas seluruh unit lapisan kedua yang menerima kontribusi $h_j^{(1)}$), lalu dikalikan dengan $\varphi'(a_j^{(1)})$. Begitu $\delta^{(1)}$ dan $\delta^{(2)}$ diketahui, seluruh entri Jacobian tinggal dibaca langsung (dengan pola yang sama seperti (13.3)–(13.5) pada Subbab 13.2, hanya saja sekarang terdapat dua pasang lapisan):

$$\begin{aligned} \frac{\partial f}{\partial w_j^{(1)}} &= \delta_j^{(1)} x, & \frac{\partial f}{\partial b_j^{(1)}} &= \delta_j^{(1)}, \\ \frac{\partial f}{\partial W_{kj}^{(2)}} &= \delta_k^{(2)} h_j^{(1)}, & \frac{\partial f}{\partial b_k^{(2)}} &= \delta_k^{(2)}, \\ \frac{\partial f}{\partial w_k^{(3)}} &= h_k^{(2)}. \end{aligned} \tag{14.6}$$

Kode MATLAB: Rekursi Sinyal Galat dan Jacobian

```
clear; clc;
x = 0.5;
w1 = [0.7; -0.4];
b1 = [0.1; 0.2];
W2 = [0.5, -0.3; 0.2, 0.6];
b2 = [0.1; -0.1];
w3 = [0.8; -0.6];
b3 = 0.05;
a1 = w1*x + b1; h1 = tanh(a1);
a2 = W2*h1 + b2; h2 = tanh(a2);
delta2 = w3 .* (1 - h2.^2);
delta1 = (W2' * delta2) .* (1 - h1.^2);
df_dw1 = delta1 * x; df_db1 = delta1;
df_dW2 = delta2 * h1'; df_db2 = delta2;
df_dw3 = h2;          df_db3 = 1;
J_row = -[df_dw1; df_db1; df_dW2(:); df_db2; df_dw3; df_db3]';
disp(J_row)
```

Catatan Aljabar Linear: Rekursi Berlaku untuk Berapa Pun Banyaknya Lapisan

Pola pada (14.5) tidak istimewa untuk dua lapisan. Pola tersebut merupakan satu langkah dari rekursi umum. Untuk jaringan dengan L lapisan tersembunyi, sinyal galat lapisan ke-l selalu dihitung dari sinyal galat lapisan ke-$(l+1)$ dengan pola yang sama:

$$\delta_j^{(l)} = \varphi'(a_j^{(l)}) \sum_k W_{kj}^{(l+1)} \, \delta_k^{(l+1)}, \qquad l = L-1, \dots, 1,$$

Perhitungan dimulai dari $\delta^{(L)}$ pada lapisan terakhir, kemudian bergerak mundur hingga $\delta^{(1)}$. Satu propagasi maju menghitung $a^{(l)}$ dan $h^{(l)}$. Sesudah itu, satu propagasi balik menghitung $\delta^{(l)}$ dan mengisi satu baris Jacobian. Pola yang sama berlaku untuk berapa pun banyaknya lapisan.

14.3 Peringkat Penuh Tidak Menjamin Kekondisian Baik

Peringkat penuh kolom menjamin bahwa $J^\top J$ invertibel secara aljabar. Namun, jaminan itu belum cukup untuk kestabilan numerik. Pada jaringan yang dalam dan sempit, J dapat berperingkat penuh tetapi memiliki bilangan kondisi yang sangat besar. Dalam aritmetika komputer, $J^\top J$ kemudian berperilaku hampir seperti matriks singular. Contoh berikut menunjukkan fenomena ini pada jaringan yang sangat kecil.

14.4 Contoh Numerik: Ketika Peringkat Penuh Saja Tidak Cukup

Tinjau kembali tugas mencocokkan $y = \sin(x)$ dengan MLP $H_1 = H_2 = 2$ (tiga belas parameter) pada 16 titik berjarak sama di interval $[-4, 4]$. Agar hasilnya benar-benar dapat diulang, parameter awal ditetapkan secara eksplisit dalam urutan $(w_1^{(1)}, w_2^{(1)}, b_1^{(1)}, b_2^{(1)}, W_{11}^{(2)}, W_{12}^{(2)}, W_{21}^{(2)}, W_{22}^{(2)}, b_1^{(2)}, b_2^{(2)}, w_1^{(3)}, w_2^{(3)}, b^{(3)})$ se-

bagai

$$\begin{aligned}\boldsymbol{\theta}_0 = (&-0.381421, -0.322414, 0.502761, -0.652934,\\ &0.160161, 0.365697, -0.499358, -0.711765,\\ &-0.360049, 0.251893, 0.099625, -0.559900, -0.107791).\end{aligned}$$

Propagasi maju memberi $S(\boldsymbol{\theta}_0) \approx 5.852$. Parameter ini sengaja dipilih untuk memperlihatkan kondisi buruk. Sinyal propagasi balik dapat mengecil walaupun matriks Jacobian tetap berperingkat penuh.

Matriks Jacobian $J \in \mathbb{R}^{16\times 13}$ memiliki $\operatorname{rank}(J) = 13$, sehingga berperingkat penuh kolom. Meskipun demikian,

$$\kappa_2(J^\top J) \approx 6.7 \times 10^{15}. \tag{14.7}$$

Bandingkan dengan diskusi bilangan kondisi pada Subbab 4.1: angka sebesar (14.7) menandakan bahwa meski $J^\top J$ *secara teoretis* invertibel, secara numerik matriks tersebut hampir tidak dapat dibedakan dari matriks singular pada presisi *floating-point* standar (≈ 16 digit). Akibatnya, langkah Gauss–Newton murni ($\lambda = 0$) benar-benar tak terkendali, seperti terlihat pada Tabel 14.1.

Langkah Eksperimen

Sebelum membaca tabel, ikuti tahapan berikut.

1. Bentuk $x_i = -4 + 8(i-1)/15$ untuk $i = 1, \dots, 16$, kemudian tetapkan $y_i = \sin(x_i)$.
2. Susun $\boldsymbol{\theta}_0$ dalam urutan yang telah ditulis. Lakukan propagasi maju dengan (14.1)–(14.3).
3. Gunakan rekursi (14.5)–(14.4) untuk mengisi Jacobian J.
4. Hitung $\operatorname{rank}(J)$ dan $\kappa_2(J^\top J)$. Selesaikan (3.13) untuk langkah Gauss–Newton tanpa redaman.
5. Ulangi dengan persamaan LM (7.1) dan $\lambda_0 = 1$. Terima langkah jika

Langkah Eksperimen *(lanjutan)*

S turun. Sesudah penerimaan, bagi λ dengan 10; sesudah penolakan, kalikan λ dengan 10.

6. Simpan S setelah delapan langkah diterima. Deret acuannya ialah 5.852, 3.611, 2.769, 2.562, 1.867, 1.704, 1.584, 1.484.

Tabel 14.1: Dampak redaman pada MLP dua lapisan tersembunyi

Langkah	$\|\boldsymbol{p}\|$	$S(\boldsymbol{\theta}_0 + \boldsymbol{p})$
Sebelum melangkah	—	5.852
Gauss–Newton murni ($\lambda = 0$)	46603	8.8×10^8 *(meledak)*
Levenberg–Marquardt ($\lambda = 1$)	1.037	3.611 *(menurun sehat)*

Norma langkah Gauss–Newton murni, $\|\boldsymbol{p}\| \approx 46603$, adalah tanda paling jelas dari matriks hampir-singular: $(J^\top J)^{-1}$ mengandung nilai eigen sekitar $1/\sigma_{\min}^2$ yang meledak begitu $\sigma_{\min} \to 0$ (bandingkan (4.3)), sehingga arah langkah yang dihasilkan hampir tak berkaitan dengan arah turun yang sesungguhnya. Menambahkan redaman $\lambda = 1$ pada (7.1) langsung memulihkan keadaan. Melanjutkan iterasi Levenberg–Marquardt adaptif (Algoritma Subbab 13.4) selama delapan langkah menghasilkan lintasan nilai objektif pada Tabel 14.2. Deret tersebut

Tabel 14.2: Lintasan nilai objektif MLP dua lapisan tersembunyi

Iterasi k	0	1	2	3	4	5	6	7
$S(\boldsymbol{\theta}_k)$	5.852	3.611	2.769	2.562	1.867	1.704	1.584	1.484

menunjukkan penurunan yang stabil. Beberapa percobaan sempat ditolak dan λ dinaikkan sebelum langkah berikutnya diterima. Perilaku ini sesuai dengan mekanisme Levenberg–Marquardt pada Subbab 13.4.

Catatan 9. *Kekondisian buruk pada contoh ini berkaitan dengan simetri ruang bobot. Menukar dua unit tersembunyi dan menyesuaikan bobot yang terhubung dapat menghasilkan fungsi f yang sama. Akibatnya, terdapat arah di ruang parameter yang hampir tidak mengubah keluaran. Arah inilah yang membuat*

$\sigma_{\min}(J)$ kecil. Fenomena tersebut memotivasi pendekatan kelengkungan terstruktur yang lebih murah dan stabil daripada menyelesaikan satu sistem besar secara langsung.

14.5 Eksperimen Publik: MLP pada Regresi Diabetes

Setelah melihat perilaku satu dan dua lapisan tersembunyi pada data sintetis, kita ulangi kedua arsitektur tersebut pada dataset publik yang sama dengan eksperimen regresi sebelumnya. Dengan demikian, perbandingan tidak berubah karena pergantian data. Yang berubah hanya bentuk modelnya.

Langkah Eksperimen

Gunakan urutan berikut agar hasil MLP dapat dibandingkan secara adil.

1. Muat dataset Diabetes, pilih BMI sebagai masukan tunggal, dan gunakan target perkembangan penyakit. Pertahankan pembagian dengan *seed* 42, yaitu 353 data latih dan 89 data uji seperti pada eksperimen regresi.

2. Standardisasi BMI dan target menggunakan rerata serta simpangan baku data latih. Sebut masukan hasil standardisasi sebagai t dan targetnya sebagai z.

3. Untuk model pertama, gunakan (13.1)–(13.2) dengan $H = 2$ dan aktivasi tanh. Urutan parameternya adalah $(w_1^{(1)}, w_2^{(1)}, b_1^{(1)}, b_2^{(1)}, w_1^{(2)}, w_2^{(2)}, b^{(2)})$. Mulai dari $(0.4, -0.3, 0.1, -0.1, 0.5, -0.4, 0)$.

4. Untuk model kedua, gunakan (14.1)–(14.3) dengan $H_1 = H_2 = 2$. Mulai dari $(0.3, -0.2, 0.1, -0.1, 0.4, 0.2, -0.3, 0.1, 0, 0, 0.5, -0.4, 0)$ dalam urutan parameter yang dijelaskan pada Subbab 14.4.

5. Pada tiap langkah, lakukan propagasi maju, hitung residu $r_i = z_i - f(t_i, \boldsymbol{\theta})$, lalu isi J dengan propagasi balik. Selesaikan persamaan LM (7.1) dengan $\lambda_0 = 10^{-2}$. Terima langkah jika S turun, kecilkan λ menjadi sepersepuluhnya, dan jika langkah ditolak, besarkan λ sepuluh kali.

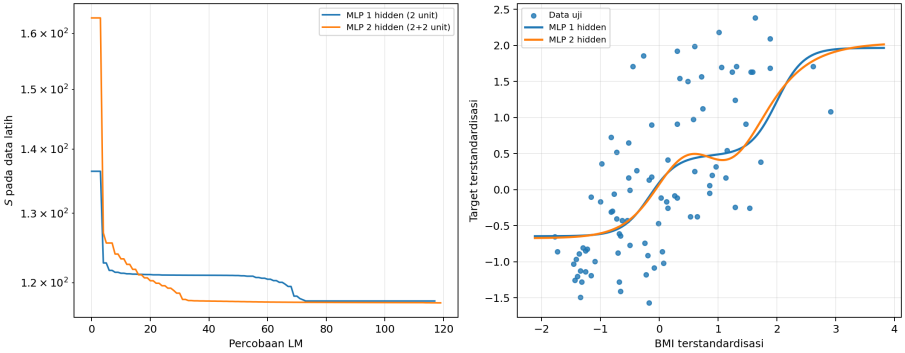
6. Jalankan hingga 60 langkah yang diterima. Simpan S pada data latih

Langkah Eksperimen *(lanjutan)*

di setiap percobaan. Kembalikan prediksi ke satuan target semula dan hitung RMSE pada data uji.

Eksperimen nyata ini menghasilkan 60 langkah diterima untuk masing-masing arsitektur. Model satu lapisan tersembunyi memerlukan 118 percobaan LM, sedangkan model dua lapisan tersembunyi memerlukan 120 percobaan. Sebagian percobaan tambahan ditolak karena nilai S tidak turun.

Gambar 14.2 memperlihatkan kurva LM dan hasil prediksi kedua arsitektur.



Gambar 14.2: Kurva LM dan hasil prediksi dua arsitektur MLP pada data Diabetes

Tabel 14.3 merangkum kinerja kedua arsitektur.

Tabel 14.3: Hasil MLP untuk regresi Diabetes

Arsitektur	Percobaan LM	S akhir	RMSE uji
MLP satu lapisan tersembunyi, 2 unit	118	117.4765	61.502
MLP dua lapisan tersembunyi, 2+2 unit	120	117.2240	60.897

Pada data ini, MLP dua lapisan tersembunyi sedikit lebih baik pada RMSE uji. Selisihnya kecil, dan hal ini juga merupakan pelajaran penting. Lapisan tambahan tidak otomatis memberi peningkatan besar, terutama ketika masukan hanya satu fitur.

14.6 MLP untuk Klasifikasi: Sigmoid dan Softmax

MLP pada contoh regresi memakai satu keluaran linear. Untuk klasifikasi, lapisan keluarannya perlu mengubah logit menjadi peluang. Bentuk umum lapisan tersembunyi pertama tetap mengikuti (13.1). Untuk K keluaran, tulis

$$\boldsymbol{z}_i = W^{(\mathrm{out})}\boldsymbol{h}_i + \boldsymbol{b}^{(\mathrm{out})}, \tag{14.8}$$

$$\boldsymbol{p}_i = \mathrm{softmax}(\boldsymbol{z}_i), \qquad K > 2, \tag{14.9}$$

sedangkan untuk klasifikasi biner digunakan $p_i = \sigma(z_i)$. Target biner ditulis $y_i \in \{0, 1\}$. Target multikelas ditulis sebagai vektor indikator $\boldsymbol{y}_i \in \{0, 1\}^K$.

Untuk menjaga hubungan dengan Gauss–Newton, kita gunakan residu kuadrat

$$\boldsymbol{r}_i = \boldsymbol{y}_i - \boldsymbol{p}_i, \qquad S(\boldsymbol{\theta}) = \tfrac{1}{2}\sum_i \|\boldsymbol{r}_i\|^2. \tag{14.10}$$

Misalkan $B_i = \partial \boldsymbol{z}_i / \partial \boldsymbol{\theta}$ adalah turunan logit terhadap seluruh parameter. Propagasi balik mengisi matriks B_i dari keluaran menuju masukan. Untuk sigmoid dan softmax, baris Jacobian residunya adalah

$$J_i^{\mathrm{bin}} = -p_i(1 - p_i)B_i, \tag{14.11}$$

$$J_i^{\mathrm{multi}} = -C_i B_i, \qquad C_i = \mathrm{diag}(\boldsymbol{p}_i) - \boldsymbol{p}_i\boldsymbol{p}_i^{\top}. \tag{14.12}$$

Persamaan (14.11)–(14.12) menunjukkan letak perbedaan antara dua tugas. Pada kasus biner, faktor keluaran hanya sebuah skalar. Pada kasus multikelas, perubahan satu logit memengaruhi seluruh peluang melalui matriks C_i. Setelah J tersusun, langkah LM tetap memakai (7.1).

14.7 Eksperimen Publik: MLP Satu dan Dua Lapisan Tersembunyi pada Klasifikasi

Sekarang kedua arsitektur MLP diuji pada klasifikasi biner dan multikelas. Studi biner memakai Breast Cancer Wisconsin Diagnostic dari UCI (Wolberg dkk.,

1993). Dataset tersebut memiliki 569 pengamatan dan 30 fitur. Agar perhitungan mudah dilacak, eksperimen menggunakan dua fitur pertama, yaitu *mean radius* dan *mean texture*. Label asli dibalik sehingga 1 menyatakan tumor ganas dan 0 menyatakan tumor jinak. Studi multikelas memakai Iris dari UCI yang telah digunakan pada Bab 12.

Langkah Eksperimen

Ikuti tahapan berikut sebelum melihat gambar dan tabel.

1. Siapkan dua dataset. Untuk tugas biner, muat Breast Cancer Wisconsin Diagnostic, ambil dua fitur pertama, dan ubah label tumor ganas menjadi 1 serta jinak menjadi 0. Dengan *seed* 42, lakukan pembagian terstratifikasi. Hasil acuan berisi 454 data latih dan 115 data uji.

2. Untuk tugas multikelas, gunakan empat fitur Iris dan tiga label kelas. Dengan *seed* 42, ambil 35 bunga per kelas untuk latih dan 15 per kelas untuk uji. Hasilnya adalah 105 data latih dan 45 data uji.

3. Hitung rerata dan simpangan baku fitur hanya dari data latih. Standardisasi data latih dan uji menggunakan nilai tersebut.

4. Bangun dua model. Model pertama memiliki empat unit pada satu lapisan tersembunyi. Model kedua memiliki empat unit pada lapisan tersembunyi pertama dan tiga unit pada lapisan tersembunyi kedua. Gunakan aktivasi tanh.

5. Untuk setiap data, lakukan propagasi maju. Hitung logit dan peluang dari (14.8)–(14.9). Bentuk residu dari (14.10). Lakukan propagasi balik untuk memperoleh B_i, lalu susun J_i dari (14.12).

6. Bentuk sistem LM (7.1) dengan $\lambda_0 = 10^{-2}$. Parameter awal ke-j ditetapkan $\theta_{0,j} = 0.1 \sin(j)$. Terima langkah jika S turun dan bagi λ dengan 10. Jika ditolak, kalikan λ dengan 10.

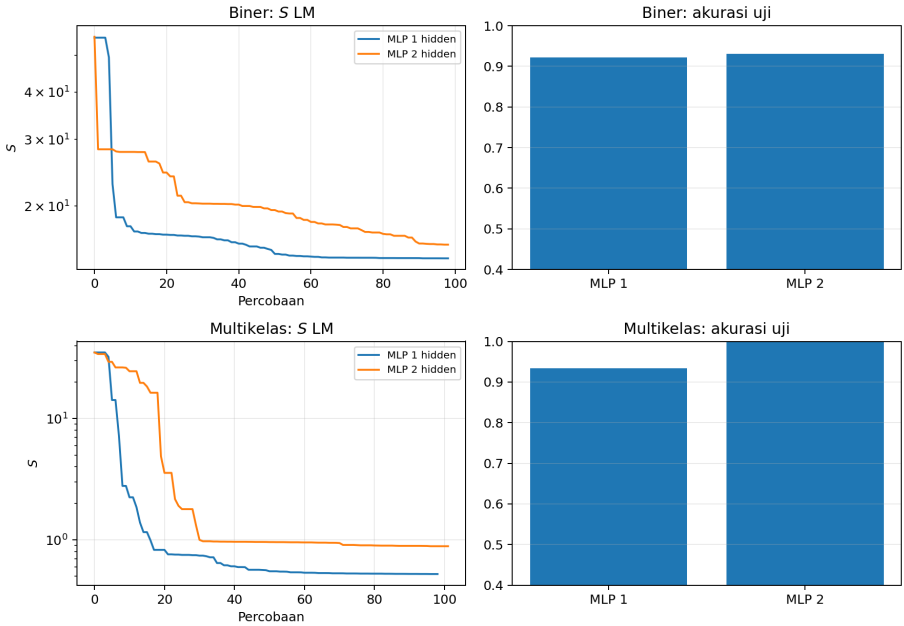
7. Jalankan sampai 50 langkah diterima. Simpan S pada setiap percobaan. Pada data uji, ubah peluang menjadi label dengan ambang 0.5 untuk

Langkah Eksperimen *(lanjutan)*

biner dan dengan mengambil peluang terbesar untuk multikelas. Hitung akurasi.

Eksperimen ini memakai residu kuadrat, bukan entropi-silang. Karena itu, nilai S antara dataset biner dan multikelas tidak dibandingkan secara langsung. Yang dibandingkan adalah perilaku kurva LM dan akurasi uji di dalam masing-masing tugas.

Gambar 14.3 memperlihatkan kurva LM dan akurasi uji kedua arsitektur.



Gambar 14.3: Kurva LM dan akurasi uji MLP pada tugas biner dan multikelas

Tabel 14.4 merangkum kinerja kedua arsitektur pada tugas biner dan multikelas.

Pada pembagian data ini, kedua arsitektur berhasil belajar klasifikasi biner dengan akurasi di atas 92%. Pada Iris, MLP dua lapisan tersembunyi mencapai

Tabel 14.4: Hasil klasifikasi MLP satu dan dua lapisan tersembunyi

Tugas dan arsitektur	Percobaan LM	S akhir	Akurasi uji
Biner, MLP satu lapisan tersembunyi	50/99	14.5462	92.17%
Biner, MLP dua lapisan tersembunyi	50/99	15.8118	93.04%
Multikelas, MLP satu lapisan tersembunyi	50/99	0.5229	93.33%
Multikelas, MLP dua lapisan tersembunyi	50/102	0.8865	100.00%

akurasi sempurna, sedangkan MLP satu lapisan tersembunyi salah mengklasifikasikan tiga bunga. Hasil ini bukan aturan umum bahwa jaringan yang lebih dalam selalu lebih baik. Hasilnya bergantung pada data, inisialisasi, dan strategi redaman.

Bab 15

Koneksi ke Teori Statistik: Matriks Informasi Fisher

▲ Pendalaman Teknis

Bab ini menjawab pertanyaan yang mungkin sudah muncul di benak sejak Bab 10: mengapa bobot $\pi_i(1-\pi_i)$ pada IRLS biner, dan $\mathrm{diag}(\boldsymbol{\pi}_i) - \boldsymbol{\pi}_i\boldsymbol{\pi}_i^\top$ pada IRLS multikelas, memiliki bentuk yang begitu rapi? Jawabannya melibatkan teori statistik (matriks informasi Fisher) yang tidak dibutuhkan untuk memakai metodenya, tetapi menjelaskan *mengapa* metodenya bekerja sebaik itu. Bab ini boleh dilewati; kita dapat langsung menuju Bab 16 atau Bab 17.

15.1 Keluarga Eksponensial dan Model Linear Umum

Sejauh ini kita telah melihat tiga bentuk fungsi kerugian: kuadrat terkecil pada Bab 3, entropi-silang biner pada Bab 10, dan entropi-silang multikelas pada Bab 12. Kemiripan ketiganya bukan kebetulan. Ketiganya merupakan kasus khusus dari keluarga distribusi peluang yang disebut **keluarga eksponensial**. Untuk peubah acak skalar y dengan parameter alami (*natural parameter*) η, bentuk umumnya adalah

$$p(y \mid \eta) = h(y)\exp\big(y\eta - A(\eta)\big), \tag{15.1}$$

dengan $h(y)$ sebagai fungsi dasar yang tidak bergantung pada η, sedangkan $A(\eta)$ disebut **fungsi partisi-log**. Model linear umum (GLM) (Nelder dan Wedderburn, 1972; McCullagh dan Nelder, 1989) memasangkan bentuk ini dengan taut kanonik $\eta_i = \boldsymbol{x}_i^\top \boldsymbol{\theta}$. Bentuk tersebut sama dengan logit z_i yang digunakan sejak Bab 10.

Dua sifat $A(\eta)$ berikut sangat berguna. Agar hubungan ini tidak diterima begitu saja, kita mulai dari syarat normalisasi $\int\, p(y \mid \eta)\, dy = 1$, lalu menurunkan kedua ruas terhadap η:

$$\begin{aligned} 0 &= \frac{d}{d\eta} \int p(y \mid \eta)\, dy = \int p(y \mid \eta) \big[y - A'(\eta) \big]\, dy \\ &= \mathbb{E}[y \mid \eta] - A'(\eta). \end{aligned}$$

Jadi, $A'(\eta) = \mathbb{E}[y \mid \eta] = \mu$. Turunkan hasil ini sekali lagi. Dengan menggunakan $\partial p / \partial \eta = p[y - A'(\eta)]$, diperoleh

$$\begin{aligned} A''(\eta) &= \frac{d}{d\eta} \int y\, p(y \mid \eta)\, dy = \int y\, p(y \mid \eta) \big[y - A'(\eta) \big]\, dy \\ &= \mathbb{E}[y^2 \mid \eta] - \mathbb{E}[y \mid \eta]^2 = \operatorname{Var}(y \mid \eta). \end{aligned}$$

Dengan demikian,

$$A'(\eta) = \mathbb{E}[y \mid \eta] =: \mu, \qquad A''(\eta) = \operatorname{Var}(y \mid \eta). \tag{15.2}$$

Turunan pertama A adalah rerata y, sedangkan turunan keduanya adalah variansi y. Dua fakta kecil ini akan membawa kita jauh.

15.2 Matriks Informasi Fisher: Definisi dan Identitas Dasarnya

Untuk model peluang $p(y \mid \boldsymbol{\theta})$ apa pun (tidak harus keluarga eksponensial), **matriks informasi Fisher** didefinisikan sebagai kovariansi dari *skor*

$\nabla_\theta \log p(y \mid \theta)$:

$$\mathcal{I}(\theta) := \mathbb{E}\Big[\nabla \log p(y \mid \theta)\, \nabla \log p(y \mid \theta)^\top\Big]. \tag{15.3}$$

Persamaan (15.3) berbentuk kovariansi. Oleh sebab itu, $\mathcal{I}(\theta)$ selalu semidefinit positif. Hubungannya dengan Hessian diberikan oleh Proposisi 4.

Proposisi 4 (Dua Wajah Informasi Fisher)**.** *Dengan syarat regularitas baku (integral dan turunan boleh dipertukarkan), berlaku*

$$\mathcal{I}(\theta) = \mathbb{E}\Big[\nabla \log p(y \mid \theta)\, \nabla \log p(y \mid \theta)^\top\Big] = -\mathbb{E}\Big[\nabla^2 \log p(y \mid \theta)\Big]. \tag{15.4}$$

Bukti. Dari $\int p(y \mid \theta)\, dy = 1$ untuk setiap θ, diferensialkan kedua ruas terhadap θ:

$$\int \nabla p(y \mid \theta)\, dy = \mathbf{0} \quad \Longrightarrow \quad \int p(y \mid \theta)\, \nabla \log p(y \mid \theta)\, dy = \mathbf{0},$$

memakai identitas $\nabla p = p\, \nabla \log p$. Hal ini berarti $\mathbb{E}[\nabla \log p(y \mid \theta)] = \mathbf{0}$. Skor memiliki rerata nol. Diferensialkan sekali lagi terhadap θ (aturan hasil kali):

$$\int p(y \mid \theta)\Big[\nabla \log p(y \mid \theta)\, \nabla \log p(y \mid \theta)^\top + \nabla^2 \log p(y \mid \theta)\Big]\, dy = \mathbf{0}.$$

Setelah diintegralkan, suku pertama di dalam kurung tidak lain adalah $\mathcal{I}(\theta)$ pada (15.3). Suku kedua adalah $\mathbb{E}[\nabla^2 \log p]$. Karena jumlah keduanya sama dengan nol, diperoleh tepat (15.4). □

Identitas ini menghubungkan kovariansi skor dengan nilai harapan Hessian log-kemungkinan. Hubungan dengan fungsi objektif buku ini perlu dinyatakan secara hati-hati. Kuadrat terkecil (1.3) setara dengan log-kemungkinan negatif Gaussian hingga konstanta dan faktor skala jika variansi galat dianggap tetap. Entropi-silang logistik dan softmax juga merupakan log-kemungkinan negatif. Sebaliknya, residu kuadrat pada klasifikasi MLP di Bab 14 dipakai sebagai pilihan optimisasi dan tidak otomatis memiliki tafsiran kemungkinan yang sama.

15.3 Mengapa IRLS Merupakan *Fisher Scoring*

Sekarang mari kita hitung skor dan Hessian untuk satu observasi keluarga eksponensial (15.1) dengan taut kanonik $\eta = \boldsymbol{x}^\top \boldsymbol{\theta}$, memakai (15.2) dan aturan rantai $\nabla_{\boldsymbol{\theta}} \eta = \boldsymbol{x}$:

$$\nabla_{\boldsymbol{\theta}} \log p(y \mid \boldsymbol{\theta}) = \big(y - A'(\eta)\big)\boldsymbol{x} = (y - \mu)\,\boldsymbol{x}, \tag{15.5}$$

$$\nabla^2_{\boldsymbol{\theta}} \log p(y \mid \boldsymbol{\theta}) = -A''(\eta)\,\boldsymbol{x}\boldsymbol{x}^\top = -\operatorname{Var}(y \mid \eta)\,\boldsymbol{x}\boldsymbol{x}^\top. \tag{15.6}$$

Catatan Aljabar Linear: Informasi Teramati Sama dengan Informasi Terekspektasi pada GLM Kanonik

Skor (15.5) masih bergantung pada y, tetapi Hessian log-kemungkinan (15.6) tidak bergantung pada realisasi y. Nilainya hanya bergantung pada η melalui $A''(\eta)$. Karena itu, untuk GLM dengan taut kanonik, informasi teramati dan informasi terekspektasi berimpit. Kesetaraan ini menjelaskan mengapa suku sisa pada Bab 10 dan 12 lenyap secara eksak.

Menjumlahkan (15.6) atas m data (dengan tanda minus, karena $S = -\sum_i \log p_i$) memberi

$$H(\boldsymbol{\theta}) = \sum_{i=1}^{m} \operatorname{Var}(y_i \mid \eta_i)\,\boldsymbol{x}_i \boldsymbol{x}_i^\top = X^\top D(\boldsymbol{\theta}) X,$$

$$D(\boldsymbol{\theta}) = \operatorname{diag}\big(\operatorname{Var}(y_1 \mid \eta_1), \dots, \operatorname{Var}(y_m \mid \eta_m)\big), \tag{15.7}$$

Bentuk $X^\top D(\boldsymbol{\theta}) X$ pada (15.7) telah muncul pada kuadrat terkecil berbobot, IRLS biner, dan IRLS multikelas. Perbedaannya terletak pada bentuk variansi yang mengisi D. Tabel 15.1 merangkum korespondensinya.

Tabel 15.1: Korespondensi keluarga eksponensial dan variansi

Distribusi y	$A(\eta)$	$A'(\eta) = \mu$	$A''(\eta) = \operatorname{Var}(y)$
Gaussian (variansi 1)	$\eta^2/2$	η	1
Bernoulli	$\log(1 + e^\eta)$	$\sigma(\eta)$	$\pi(1-\pi)$

Baris pertama menjelaskan bobot $D = I$ pada kuadrat terkecil dengan variansi Gaussian konstan. Baris kedua menghasilkan bobot regresi logistik pada Bab 10. Dalam *Fisher scoring*, Hessian teramati diganti dengan informasi Fisher. Untuk GLM dengan taut kanonik yang dibahas di sini, keduanya berimpit. Karena itu, langkah *Fisher scoring* sama dengan langkah Newton eksak. Kesetaraan ini bergantung pada struktur model dan tidak boleh digeneralisasi ke semua jaringan saraf.

Catatan 10. *Pendekatan* Generalized Gauss–Newton *(GGN) pada MLP mempertahankan kelengkungan fungsi kerugian terhadap keluaran, lalu memetakannya ke ruang parameter melalui Jacobian. Untuk distribusi keluaran tertentu dari keluarga eksponensial dan parameterisasi yang sesuai, GGN dapat berimpit dengan informasi Fisher model. Di luar syarat tersebut, keduanya hanya berkaitan dan tidak harus identik. Gradien alami (Amari, 1998) menggunakan informasi Fisher sebagai metrik, sedangkan langkah GGN menggunakan matriks kelengkungan GGN. Arah yang dihasilkan dapat sama pada kasus khusus, tetapi kesamaan itu bukan sifat universal (Martens, 2020).*

Bab 16

Konvergensi Global dengan Pencarian Garis Armijo–Wolfe

▲ Pendalaman Teknis

Teorema 1 pada Bab 6 hanya menjamin konvergensi *lokal*. Artinya, jaminan tersebut berlaku jika iterasi dimulai cukup dekat dengan $\boldsymbol{\theta}^*$. Bagaimana jika tebakan awal jauh dari solusi? Pertanyaan ini penting karena contoh MLP pada Subbab 13.5 dan 14.4 menunjukkan bahwa langkah mentah dapat meledak. Bab ini memberikan jawabannya melalui pencarian garis.

16.1 Mengapa Konvergensi Lokal Saja Tidak Cukup

Ingat kembali algoritma dasar Subbab 3.4. Pada setiap iterasi, arah $\boldsymbol{p}_k$ dihitung lebih dahulu, kemudian parameter diperbarui melalui $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha_k \boldsymbol{p}_k$. Nilai $\alpha_k = 1$ tidak selalu aman. Contoh pada Subbab 13.5 dan 14.4 menunjukkan bahwa langkah penuh dapat menaikkan fungsi objektif secara drastis. Pertanyaan yang tepat bukan apakah satu aturan bekerja dari semua titik awal. Pertanyaannya ialah syarat apa yang membuat setiap langkah memberi kemajuan dan, bersama asumsi kemulusan serta keterbatasan, menghasilkan barisan bagian dengan gradien menuju nol.

16.2 Syarat Armijo dan Wolfe

Syarat pertama, disebut **syarat Armijo** (penurunan cukup) (Armijo, 1966), menuntut α_k membuat S turun *sebanding* dengan penurunan yang diprediksi model linear, bukan sekadar turun sedikit saja:

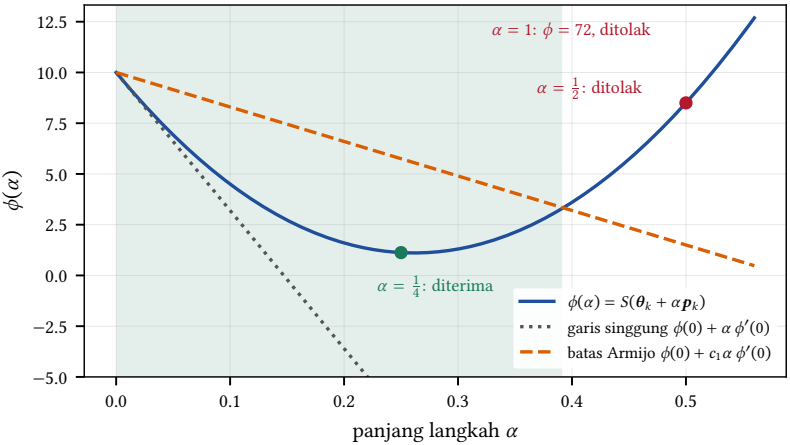
$$S(\boldsymbol{\theta}_k + \alpha_k \boldsymbol{p}_k) \le S(\boldsymbol{\theta}_k) + c_1 \alpha_k \nabla S(\boldsymbol{\theta}_k)^{\top} \boldsymbol{p}_k, \qquad c_1 \in (0,1), \qquad \text{biasanya } c_1 = 10^{-4}. \tag{16.1}$$

Kode MATLAB: *Backtracking Line Search* (Syarat Armijo)

```
clear; clc;
S = @(t) 0.5*((t(1)-1)^2 + 4*(t(2)+2)^2);
theta = [3; 0];
grad = [theta(1)-1; 4*(theta(2)+2)];
p = -grad;
alpha = 1; c1 = 1e-4;
S0 = S(theta); gTp = grad'*p;
while S(theta + alpha*p) > S0 + c1*alpha*gTp
    alpha = 0.5 * alpha;
end
theta = theta + alpha*p;
disp(alpha)
disp(theta)
```

Gambar 16.1 memvisualisasikan syarat (16.1) untuk fungsi pada kotak kode di atas, dengan $\boldsymbol{\theta}_k = (3, 0)$ dan $\boldsymbol{p}_k = -\nabla S(\boldsymbol{\theta}_k)$. Agar garis batasnya terlihat jelas, gambar memakai $c_1 = 0.25$. Daerah berwarna menandai nilai α yang memenuhi syarat Armijo. Pencarian garis mundur mencoba $\alpha = 1$ ($\phi = 72$) dan $\alpha = \frac{1}{2}$ ($\phi = 8.5$), keduanya ditolak, lalu menerima $\alpha = \frac{1}{4}$ ($\phi = 1.125$). Dengan $c_1 = 10^{-4}$ seperti pada kotak kode, batasnya hampir berimpit dengan $\phi(0)$ sehingga $\alpha = \frac{1}{2}$ sudah diterima.

Karena $\boldsymbol{p}_k$ arah turun (Proposisi 1), ruas kanan (16.1) selalu lebih kecil dari $S(\boldsymbol{\theta}_k)$ untuk $\alpha_k > 0$ kecil. Jadi, syarat ini selalu dapat dipenuhi dengan memperkecil α_k (inilah dasar *backtracking line search* yang sudah berulang kali disebut di algoritma-algoritma sebelumnya: coba $\alpha = 1$, jika (16.1) gagal,



Gambar 16.1: Syarat Armijo pada fungsi satu peubah $\phi(\alpha) = S(\boldsymbol{\theta}_k + \alpha \boldsymbol{p}_k)$ dan hasil pencarian garis mundur

kecilkan α, ulangi). Syarat kedua, **syarat Wolfe** (kelengkungan) (Wolfe, 1969), mencegah α_k terlalu kecil:

$$\nabla S(\boldsymbol{\theta}_k + \alpha_k \boldsymbol{p}_k)^\top \boldsymbol{p}_k \geq c_2 \, \nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k, \qquad c_2 \in (c_1, 1). \tag{16.2}$$

Kedua syarat tersebut mencegah α_k terlalu besar maupun terlalu kecil. Di bawah asumsi baku bahwa S terbatas di bawah sepanjang arah pencarian, terdapat $\alpha_k > 0$ yang memenuhi keduanya (Nocedal dan Wright, 2006).

16.3 Teorema Zoutendijk dan Jaminan Konvergensi Global

Teorema 3 (Zoutendijk). *Misalkan S terbatas di bawah dan ∇S Lipschitz kontinu dengan konstanta L. Jika setiap $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha_k \boldsymbol{p}_k$ memakai $\boldsymbol{p}_k$ arah turun dan α_k memenuhi syarat Armijo–Wolfe (16.1)–(16.2), maka*

$$\sum_{k=0}^{\infty} \cos^2 \psi_k \, \|\nabla S(\boldsymbol{\theta}_k)\|^2 < \infty, \qquad \cos \psi_k := \frac{-\nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k}{\|\nabla S(\boldsymbol{\theta}_k)\| \, \|\boldsymbol{p}_k\|}. \tag{16.3}$$

Teorema ini berasal dari Zoutendijk (1970). Buktinya (baku dalam literatur optimisasi numerik, memakai syarat Wolfe untuk membangun deret teleskopik yang dikendalikan oleh kekontinuan Lipschitz ∇S) ada di Lampiran B. Yang lebih penting bagi kita adalah akibatnya:

Proposisi 5 (Konvergensi Global Gauss–Newton dengan Armijo–Wolfe)**.** *Misalkan $\sigma_{\min}(J_k) \geq \sigma_0 > 0$ dan $\|J_k\| \leq M_0$ untuk semua k (bilangan kondisi J_k terbatas seragam). Dengan asumsi tersebut, $\cos\psi_k \geq \delta := \sigma_0^2/M_0^2 > 0$ untuk semua k, sehingga deret (16.3) memaksa*

$$\liminf_{k\to\infty} \|\nabla S(\boldsymbol{\theta}_k)\| = 0.$$

Bukti. Untuk arah Gauss–Newton $\boldsymbol{p}_k = -(J_k^\top J_k)^{-1} J_k^\top \boldsymbol{r}_k = -(J_k^\top J_k)^{-1}\nabla S(\boldsymbol{\theta}_k)$ (memakai (2.4)), kosinus sudutnya adalah

$$\begin{aligned}
\cos\psi_k &= \frac{\nabla S(\boldsymbol{\theta}_k)^\top (J_k^\top J_k)^{-1} \nabla S(\boldsymbol{\theta}_k)}{\|\nabla S(\boldsymbol{\theta}_k)\|\,\|(J_k^\top J_k)^{-1}\nabla S(\boldsymbol{\theta}_k)\|} \\
&\geq \frac{\sigma_{\min}\!\left((J_k^\top J_k)^{-1}\right)\|\nabla S(\boldsymbol{\theta}_k)\|^2}{\|\nabla S(\boldsymbol{\theta}_k)\|\cdot\sigma_{\max}\!\left((J_k^\top J_k)^{-1}\right)\|\nabla S(\boldsymbol{\theta}_k)\|} \\
&= \frac{\sigma_{\min}(J_k^\top J_k)}{\sigma_{\max}(J_k^\top J_k)} \;\geq\; \frac{\sigma_0^2}{M_0^2},
\end{aligned}$$

memakai sifat baku nilai eigen matriks simetris definit positif dan definisi bilangan kondisi (4.3) dari Subbab 4.1. Karena $\cos\psi_k \geq \delta > 0$ seragam, (16.3) menjadi $\delta^2 \sum_k \|\nabla S(\boldsymbol{\theta}_k)\|^2 < \infty$, yang mengharuskan $\|\nabla S(\boldsymbol{\theta}_k)\| \to 0$ sepanjang suatu barisan bagian (jika tidak, sukunya tidak menuju nol dan deretnya divergen). □

Catatan 11. *Proposisi 5 menjelaskan secara formal kegagalan pada Subbab 13.5 dan 14.4. Pada kedua contoh tersebut, $\alpha_k = 1$ dipaksakan tanpa memeriksa syarat Armijo–Wolfe. Jika pencarian garis mundur digunakan, langkah yang menaikkan S dari* 1.339 *menjadi* 802.9 *tidak akan diterima. Algoritma akan mengecilkan α sampai (16.1) terpenuhi. Efeknya mirip dengan Levenberg–Marquardt, tetapi mekanismenya berbeda. Pencarian garis memperpendek langkah pada arah yang*

sama, sedangkan Levenberg–Marquardt juga dapat mengubah arah. Kedua pendekatan saling melengkapi. Levenberg–Marquardt membantu ketika $J^\top J$ mendekati singular, sedangkan syarat Armijo–Wolfe menjaga panjang langkah agar tetap aman.

Proposisi 5 hanya menjamin $\liminf \|\nabla S(\boldsymbol{\theta}_k)\| = 0$. *Artinya, terdapat barisan bagian yang menuju titik stasioner, tetapi belum tentu seluruh barisan memenuhi* $\boldsymbol{\theta}_k \to \boldsymbol{\theta}^*$. *Konvergensi penuh memerlukan syarat tambahan, misalnya keunikan titik stasioner atau kekompakan himpunan level* $\{\boldsymbol{\theta} : S(\boldsymbol{\theta}) \le S(\boldsymbol{\theta}_0)\}$. *Setelah iterasi cukup dekat dengan* $\boldsymbol{\theta}^*$, *Teorema 1 pada Bab 6 mengambil alih dan menentukan laju konvergensinya. Dengan demikian, hasil global membawa iterasi menuju lingkungan solusi, sedangkan hasil lokal menjelaskan kecepatannya di dalam lingkungan tersebut.*

Bab 17

Ringkasan Sifat-Sifat Penting

Setelah menelusuri seluruh gagasan dan turunannya, kita dapat merangkum perjalanan buku ini dalam beberapa kesimpulan utama.

17.1 Peta Keputusan untuk Memilih Metode

Ringkasan paling berguna bukan hanya daftar rumus, tetapi peta keputusan. Mulailah dari bentuk keluaran dan struktur model. Setelah itu, periksa kondisi Jacobian dan perilaku nilai objektif selama iterasi.

Peta pemilihan metode yang dirangkum pada Tabel 17.1 dapat dipakai sebagai pemeriksaan awal ketika menghadapi masalah baru.

Tabel 17.1: Peta pemilihan metode berdasarkan masalah

Situasi	Metode awal	Pemeriksaan lanjutan
Regresi nonlinear yang stabil	Gauss–Newton	Pastikan J berperingkat penuh dan langkah menurunkan S
Regresi nonlinear yang tidak stabil	QR atau SVD, lalu LM bila perlu	Pantau nilai singular, kondisi, dan redaman
Data dengan ketidakpastian berbeda	Kuadrat terkecil berbobot	Verifikasi sumber bobot dan skala residu
Klasifikasi satu lapisan	IRLS biner atau softmax multikelas	Periksa separasi dan gunakan satu kelas sebagai referensi
Jaringan dengan lapisan tersembunyi	*Generalized Gauss–Newton* atau LM	Hitung Jacobian melalui propagasi balik dan simpan kurva pembelajaran

Tabel 17.1 dapat dibaca dari atas ke bawah ketika menghadapi masalah baru. Tabel ini tidak menggantikan analisis data, tetapi membantu menentukan pemeriksaan pertama yang paling masuk akal.

17.2 Kesimpulan Teoretis

- Gauss–Newton mengaproksimasi Hessian dengan matriks Gram $J^\top J$, sehingga hanya memerlukan turunan pertama (Jacobian) tanpa turunan kedua.
- $J^\top J$ selalu simetris dan semidefinit positif untuk sembarang J; matriks ini menjadi definit positif (dan karenanya invertibel) tepat ketika J berperingkat penuh kolom.
- Persamaan normal $J^\top J\,\boldsymbol{p} = -J^\top \boldsymbol{r}$ pada dasarnya setara dengan proyeksi ortogonal arah koreksi $-\boldsymbol{r}$ ke ruang kolom J.
- Arah langkah $\boldsymbol{p}_k$ selalu merupakan arah turun bila J_k berperingkat penuh kolom, sehingga kemajuan di tiap iterasi terjamin.
- Secara numerik, selesaikan subproblem melalui QR atau SVD. Pembentukan $J^\top J$ secara eksplisit mengkuadratkan bilangan kondisi.

- Konvergensi bersifat kuadratik pada kasus residu nol di solusi, dan menurun menjadi linear (dengan rasio κ) seiring membesarnya residu dan kelengkungan model relatif terhadap $\sigma_{\min}(J(\boldsymbol{\theta}^*))$.

- Levenberg–Marquardt muncul secara alami dari kondisi KKT pada subproblem daerah kepercayaan Gauss–Newton. Metode ini mengurangi ketidakstabilan numerik dan risiko divergensi ketika $J^\top J$ mendekati singular. Parameter λ_k dapat disesuaikan dengan faktor tetap Marquardt atau rasio keuntungan Nielsen. Keduanya memiliki jaminan konvergensi global pada Teorema 2 dan kembali ke laju Gauss–Newton ketika $\lambda_k \to 0$. Pada eksperimen buku ini, Nielsen memerlukan lebih sedikit percobaan langkah. Metode dogleg dan persamaan sekuler Moré melengkapi kedua strategi tersebut dengan mengendalikan radius Δ_k secara langsung.

- Generalisasi ke kuadrat terkecil berbobot ($J^\top W J\, \boldsymbol{p} = -J^\top W \boldsymbol{r}$) memungkinkan metode ini menangani heteroskedastisitas, dan berkaitan langsung dengan estimasi kemungkinan maksimum di bawah asumsi galat Gaussian.

- Gagasan yang sama juga menjangkau klasifikasi biner melalui regresi logistik: metode Newton untuk kerugian entropi-silang berbentuk persis persamaan normal berbobot ($X^\top D(\boldsymbol{\theta}) X\, \boldsymbol{p} = -X^\top(\boldsymbol{\pi} - \boldsymbol{y})$), dan karena logitnya linear terhadap parameter, aproksimasi Gauss–Newton di sini bahkan sama persis dengan Hessian eksak. Algoritma ini dikenal sebagai IRLS.

- Perluasan ke K kelas melalui softmax pada perceptron satu lapisan mewarisi seluruh struktur ini: gradien dan Hessian memiliki bentuk yang sama ($\widetilde{X}^\top \mathcal{D}(\boldsymbol{\theta}) \widetilde{X}\, \boldsymbol{p} = -\widetilde{X}^\top(\boldsymbol{\pi} - \boldsymbol{y})$), hanya "bobot" skalar berubah menjadi blok matriks kovariansi kategorikal $\operatorname{diag}(\boldsymbol{\pi}_i) - \boldsymbol{\pi}_i \boldsymbol{\pi}_i^\top$ yang tetap semidefinit positif dan tetap membuat Gauss–Newton sama persis dengan Newton eksak, selama modelnya satu lapisan linear.

- Begitu ditambahkan lapisan tersembunyi dengan aktivasi nonlinear (MLP), keistimewaan "Gauss–Newton = Newton eksak" hilang: suku $M(\boldsymbol{\theta})$ pada

dekomposisi Hessian tidak lagi nol (kecuali untuk aktivasi *piecewise-linear* seperti ReLU). Propagasi balik pada dasarnya adalah cara menghitung matriks Jacobian $J(\boldsymbol{\theta})$ ini, dan karena J kini berubah nonlinear terhadap $\boldsymbol{\theta}$, langkah Gauss–Newton murni tanpa redaman dapat melenceng jauh. Hal ini menjadikan Levenberg–Marquardt (Bab 7) kebutuhan praktis, bukan sekadar pelengkap teoretis.

- Pada jaringan dengan dua atau lebih lapisan, propagasi balik berwujud rekursi sinyal galat $\delta^{(l)}$ dari lapisan keluaran menuju lapisan masukan. Pola ini berlaku untuk berapa pun banyaknya lapisan. Jaringan yang sempit dan dalam sering memiliki Jacobian yang kekurangan peringkat karena simetri dan redundansi parameter. Dalam keadaan tersebut, regularisasi LM menjadi kebutuhan, bukan sekadar pilihan tambahan.

17.3 Kesalahan yang Sering Terjadi

Ada beberapa kegagalan yang berulang dalam implementasi Gauss–Newton. Kesalahan pertama adalah membentuk $J^\top J$ lalu menyelesaikannya dengan invers eksplisit. Cara itu tampak singkat di atas kertas, tetapi dapat memperburuk kekondisian dan mengubah galat pembulatan menjadi langkah yang sangat besar. Gunakan QR atau SVD untuk implementasi yang serius.

Kesalahan kedua adalah mencampur definisi residu. Jika buku atau program memakai $r_i = y_i - f_i$, maka gradiennya adalah $J^\top r$. Jika dipakai $r_i = f_i - y_i$, tanda Jacobian atau tanda ruas kanan harus diubah secara konsisten. Perbedaan tanda yang kecil di satu baris dapat membuat seluruh iterasi bergerak ke arah yang salah.

Kesalahan ketiga adalah membandingkan metode dengan kondisi eksperimen yang berbeda. Tebakan awal, pembagian data, skala fitur, kriteria berhenti, dan jumlah percobaan harus dibuat sama. Tanpa keseragaman ini, kurva pembelajaran tidak dapat digunakan untuk menyimpulkan bahwa satu metode lebih baik daripada metode lain.

17.4 Prosedur Praktis dari Awal sampai Akhir

Berikut ini adalah alur kerja yang dapat dipakai untuk hampir semua studi kasus dalam buku ini.

1. Nyatakan bentuk keluaran. Tentukan apakah masalahnya regresi, klasifikasi biner, atau klasifikasi multikelas.
2. Tulis fungsi model dan fungsi kerugian sebelum menulis program.
3. Tentukan residu, Jacobian, dan gradien. Periksa dimensinya dengan satu contoh kecil.
4. Bagi data menjadi data latih dan data uji. Simpan aturan pembagian agar percobaan dapat diulang.
5. Standardisasikan fitur jika rentang antarfitur sangat berbeda.
6. Pilih penyelesaian QR atau SVD. Gunakan Levenberg–Marquardt jika langkah Gauss–Newton tidak stabil.
7. Simpan nilai objektif, norma gradien, norma langkah, dan metrik uji pada setiap iterasi.
8. Sajikan kurva pembelajaran serta tabel hasil akhir. Jelaskan juga kegagalan atau langkah yang ditolak.

Alur ini mengubah metode numerik menjadi eksperimen yang dapat dipertanggungjawabkan. Kita tidak hanya memperoleh satu angka akurasi, tetapi juga dapat menelusuri bagaimana angka itu diperoleh dan mengapa algoritma berhenti.

17.5 Penutup

Gagasan utama buku ini dapat diringkas dalam satu kalimat. Gauss–Newton menggunakan informasi kemiringan melalui Jacobian untuk membangun mo-

del kuadrat yang murah, lalu memperbaiki langkahnya dengan pemeriksaan kestabilan ketika model lokal itu tidak lagi dapat dipercaya.

Dari titik awal yang sederhana, gagasan tersebut berkembang secara alami. SVD menjelaskan kapan sistem linear aman. Levenberg–Marquardt mengendalikan langkah ketika sistem mendekati singular. IRLS menunjukkan bahwa struktur yang sama muncul pada klasifikasi. Softmax memperluasnya ke banyak kelas. Propagasi balik membawa Jacobian ke jaringan saraf berlapis. Akhirnya, teori konvergensi dan eksperimen publik menghubungkan rumus dengan perilaku yang dapat diamati.

Setelah menyelesaikan latihan dan eksperimen pada bab-bab sebelumnya, kita sudah memiliki bekal untuk membaca pustaka optimisasi numerik yang lebih besar. Langkah berikutnya bukan menghafal lebih banyak rumus, melainkan memilih model yang sesuai, memeriksa asumsi, dan melaporkan hasil dengan jujur.

Bab 18

Latihan Bertahap

Bab ini dirancang sebagai jembatan dari membaca ke mengerjakan. Kerjakan soal tingkat dasar terlebih dahulu. Jika langkahnya sudah terasa alami, lanjutkan ke tingkat menengah dan penerapan. Tidak semua soal membutuhkan pemrograman; sebagian sengaja dibuat kecil agar dapat dihitung dengan tangan.

18.1 Cara Menggunakan Latihan

Jangan langsung melihat jawaban dari program. Untuk setiap soal, tulis lebih dahulu ukuran setiap vektor dan matriks. Setelah itu, hitung satu langkah secara manual atau dengan kalkulator. Barulah gunakan MATLAB atau Octave untuk memeriksa hasilnya. Kebiasaan ini membantu membedakan kesalahan pada rumus, kesalahan dimensi, dan kesalahan pengetikan.

Pada soal penerapan, simpan sedikitnya empat keluaran: nilai fungsi sebelum langkah, nilai fungsi sesudah langkah, norma gradien, dan norma langkah. Empat angka ini cukup untuk menjawab pertanyaan dasar apakah algoritma bergerak ke arah yang benar.

Latihan Bertahap Tingkat Dasar

1. Diberikan data $(x, y) = (0, 1), (1, 3)$ dan model $f(x, \theta) = \theta_1 + \theta_2 x$. Hitung kedua residu ketika $\boldsymbol{\theta} = (1, 1)$.
2. Dengan residu pada soal 1, tuliskan vektor $\boldsymbol{r}$ dan nilai $S(\boldsymbol{\theta}) = \frac{1}{2}\|\boldsymbol{r}\|^2$.
3. Untuk model yang sama, tuliskan Jacobian residu terhadap (θ_1, θ_2). Periksa bahwa setiap baris berkaitan dengan satu data.

Petunjuk: turunkan $r_i = y_i - \theta_1 - \theta_2 x_i$ terhadap kedua parameter.

Latihan Bertahap Tingkat Menengah

1. Gunakan
$$J = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}, \qquad \boldsymbol{r} = \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}.$$
Hitung $J^\top J$, $J^\top \boldsymbol{r}$, lalu selesaikan persamaan normal untuk $\boldsymbol{p}$.
2. Periksa secara langsung bahwa $J^\top J$ pada soal 1 simetris dan definit positif.
3. Untuk $J = \begin{pmatrix} 1 & 0 \\ 0 & 2 \\ 0 & 0 \end{pmatrix}$, hitung bilangan kondisi J dan $J^\top J$. Jelaskan mengapa QR lebih aman daripada membentuk matriks Gram.

Petunjuk: untuk matriks 2×2, definit positif dapat diperiksa melalui diagonal positif dan determinan positif.

Latihan Bertahap Tingkat Penerapan

1. Gunakan data klasifikasi $(x, y) = (-1, 0), (1, 1)$ dengan model logistik $\pi(x) = \sigma(\theta_0 + \theta_1 x)$ dan tebakan awal $\boldsymbol{\theta}_0 = (0, 0)$. Hitung peluang awal, gradien, dan satu langkah IRLS.
2. Jelaskan apa yang terjadi pada langkah Levenberg–Marquardt jika λ

Latihan Bertahap Tingkat Penerapan *(lanjutan)*

dibuat sangat kecil dan jika λ dibuat sangat besar.

3. Buat program MATLAB atau Octave yang membandingkan satu iterasi Gauss–Newton dan Levenberg–Marquardt pada model $f(x, \boldsymbol{\theta}) = \theta_1 e^{\theta_2 x}$. Catat residu, Jacobian, arah langkah, dan nilai fungsi sebelum serta sesudah pembaruan.

Petunjuk: mulai dari tebakan $\boldsymbol{\theta} = (1, 0.5)$ dan gunakan QR jika sistem normal terasa tidak stabil.

Intuisi

Jika kita dapat menjelaskan dengan kata-kata mengapa Jacobian, gradien, dan matriks $J^\top J$ muncul, berarti kita sudah memahami struktur utama metode ini. Perhitungan yang lebih panjang hanyalah penerapan berulang dari tiga gagasan tersebut.

18.2 Contoh Format Jawaban yang Baik

Jawaban yang baik tidak hanya menuliskan angka akhir. Untuk satu iterasi Gauss–Newton, tuliskan urutannya sebagai berikut:

1. nyatakan tebakan awal $\boldsymbol{\theta}_0$;
2. hitung prediksi dan residu $\boldsymbol{r}_0$;
3. tulis Jacobian J_0 beserta ukurannya;
4. hitung atau faktorkan sistem untuk memperoleh $\boldsymbol{p}_0$;
5. perbarui $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_0 + \alpha_0 \boldsymbol{p}_0$;
6. bandingkan $S(\boldsymbol{\theta}_1)$ dengan $S(\boldsymbol{\theta}_0)$.

Jika S tidak turun, jangan langsung menyimpulkan bahwa turunannya salah. Periksa dahulu apakah $\alpha_0 = 1$ terlalu besar. Lakukan pencarian garis sederhana

dengan $\alpha_0 = 1/2$, kemudian $1/4$ jika diperlukan. Dengan format ini, hasil latihan menjadi catatan eksperimen kecil yang dapat dibaca ulang.

Lampiran A

Bukti Lengkap Teorema Konvergensi Lokal

Lampiran ini memuat bukti lengkap Teorema 1 (Subbab 6.3), yang pada badan teks digantikan dengan sketsa singkat agar alur baca utama tetap ringan bagi yang belum terbiasa dengan estimasi Lipschitz dan ekspansi Taylor terintegralkan.

Bukti Teorema 1. Tulis $\boldsymbol{e}_k = \boldsymbol{\theta}_k - \boldsymbol{\theta}^*$. Dari iterasi Gauss–Newton murni, kita memiliki

$$\boldsymbol{e}_{k+1} = \boldsymbol{\theta}_k - \boldsymbol{\theta}^* - \left(J_k^\top J_k\right)^{-1} J_k^\top \boldsymbol{r}_k = \left(J_k^\top J_k\right)^{-1} \left[J_k^\top J_k\, \boldsymbol{e}_k - J_k^\top \boldsymbol{r}_k\right]. \tag{A.1}$$

Langkah 1 (Ekspansi Taylor residu). Karena $\boldsymbol{r}$ terdiferensialkan kontinu, teorema nilai rata-rata terintegralkan memberi

$$\boldsymbol{r}_k = \boldsymbol{r}(\boldsymbol{\theta}^* + \boldsymbol{e}_k) = \boldsymbol{r}(\boldsymbol{\theta}^*) + \int_0^1 J(\boldsymbol{\theta}^* + t\boldsymbol{e}_k)\, dt\ \boldsymbol{e}_k. \tag{A.2}$$

Karena $J(\boldsymbol{\theta}^*)^\top \boldsymbol{r}(\boldsymbol{\theta}^*) = \mathbf{0}$ (kondisi stasioner (2.5)), substitusi (A.2) lalu pengelompokan ulang memberikan

$$J_k^\top \boldsymbol{r}_k = \underbrace{\big(J_k - J(\boldsymbol{\theta}^*)\big)^\top \boldsymbol{r}(\boldsymbol{\theta}^*)}_{=:\,\boldsymbol{a}_k} + J_k^\top J(\boldsymbol{\theta}^*)\boldsymbol{e}_k + J_k^\top \left[\int_0^1 J(\boldsymbol{\theta}^* + t\boldsymbol{e}_k)\,dt - J(\boldsymbol{\theta}^*)\right]\boldsymbol{e}_k. \tag{A.3}$$

Langkah 2 (Substitusi). Kita akan menyubstitusikan (A.3) ke dalam kurung siku pada (A.1). Dengan menuliskan $J_k^\top J_k\,\boldsymbol{e}_k = J_k^\top J(\boldsymbol{\theta}^*)\boldsymbol{e}_k + J_k^\top (J_k - J(\boldsymbol{\theta}^*))\boldsymbol{e}_k$, kita peroleh

$$J_k^\top J_k\,\boldsymbol{e}_k - J_k^\top \boldsymbol{r}_k = -\boldsymbol{a}_k + J_k^\top (J_k - J(\boldsymbol{\theta}^*))\boldsymbol{e}_k - J_k^\top \left[\int_0^1 J(\boldsymbol{\theta}^* + t\boldsymbol{e}_k)\,dt - J(\boldsymbol{\theta}^*)\right]\boldsymbol{e}_k. \tag{A.4}$$

Suku $\boldsymbol{a}_k$ ini berkaitan langsung dengan $M(\boldsymbol{\theta}^*)$: untuk residu yang mulus, berlaku $\boldsymbol{a}_k \approx M(\boldsymbol{\theta}^*)\boldsymbol{e}_k + O(\|\boldsymbol{e}_k\|^2)$, sehingga $\|\boldsymbol{a}_k\| \le \sigma\kappa\|\boldsymbol{e}_k\| + O(\|\boldsymbol{e}_k\|^2)$.

Langkah 3 (Estimasi Lipschitz). Kita membatasi setiap suku pada (A.4). Karena $\|J_k - J(\boldsymbol{\theta}^*)\| \le \gamma\|\boldsymbol{e}_k\|$ dan $\left\|\int_0^1 J(\boldsymbol{\theta}^* + t\boldsymbol{e}_k)dt - J(\boldsymbol{\theta}^*)\right\| \le \frac{\gamma}{2}\|\boldsymbol{e}_k\|$, serta $\|J_k\| \le \|J(\boldsymbol{\theta}^*)\| + \gamma\|\boldsymbol{e}_k\|$, dengan menggabungkan suku dominan orde $\|\boldsymbol{e}_k\|$ dan $\|\boldsymbol{e}_k\|^2$ kita peroleh

$$\left\|J_k^\top J_k\,\boldsymbol{e}_k - J_k^\top \boldsymbol{r}_k\right\| \le \left[\sigma\kappa + \tfrac{3}{2}\gamma\|J(\boldsymbol{\theta}^*)\|\,\|\boldsymbol{e}_k\|\right]\|\boldsymbol{e}_k\| + O(\|\boldsymbol{e}_k\|^3). \tag{A.5}$$

Langkah 4 (Mengontrol $(J_k^\top J_k)^{-1}$). Untuk $\|\boldsymbol{e}_k\|$ yang cukup kecil sehingga $\gamma\|\boldsymbol{e}_k\| < \sigma$, kekontinuan Lipschitz nilai singular memberi $\sigma_{\min}(J_k) \ge \sigma - \gamma\|\boldsymbol{e}_k\| > 0$, sehingga

$$\left\|\left(J_k^\top J_k\right)^{-1}\right\| \le \frac{1}{(\sigma - \gamma\|\boldsymbol{e}_k\|)^2} \le \frac{1}{(1-\kappa)\sigma^2}. \tag{A.6}$$

Langkah 5 (Menggabungkan semuanya). Menyisipkan (A.5) dan (A.6) ke dalam (A.1) memberikan tepat bentuk (6.4) setelah konstantanya disederhanakan. Karena $\kappa < 1$, faktor $[\kappa + c\|\boldsymbol{e}_k\|]$ akan bernilai kurang dari 1 untuk $\boldsymbol{e}_k$ cukup kecil, sehingga $\|\boldsymbol{e}_k\|$ menurun secara monoton dan barisan tetap berada di dalam bola $\mathcal{B}(\boldsymbol{\theta}^*, \bar{\rho})$ untuk $\bar{\rho}$ yang cukup kecil (melalui argumen induksi standar), lalu konvergen ke $\boldsymbol{\theta}^*$.

Jika $\kappa = 0$, suku pertama pada (6.4) lenyap dan kita peroleh $\|\boldsymbol{e}_{k+1}\| \leq C\|\boldsymbol{e}_k\|^2$. Inilah definisi konvergensi kuadratik. Jika $\kappa > 0$, untuk $\boldsymbol{e}_k$ yang kecil, suku yang mendominasi adalah $\kappa\|\boldsymbol{e}_k\|$, sehingga konvergensinya linear dengan rasio κ. □

Lampiran B

Bukti Teorema Zoutendijk

Lampiran ini memuat bukti Teorema 3 (Subbab 16.3), yang pada badan teks hanya dinyatakan tanpa bukti agar alur baca utama tetap ringan.

Bukti Teorema 3. Dari syarat Wolfe (16.2), kurangkan $\nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k$ dari kedua ruas:

$$\left[\nabla S(\boldsymbol{\theta}_{k+1}) - \nabla S(\boldsymbol{\theta}_k)\right]^\top \boldsymbol{p}_k \geq (c_2 - 1)\, \nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k.$$

Karena ∇S Lipschitz dengan konstanta L, ruas kiri terbatas di atas oleh $\|\nabla S(\boldsymbol{\theta}_{k+1}) - \nabla S(\boldsymbol{\theta}_k)\|\,\|\boldsymbol{p}_k\| \leq L\alpha_k \|\boldsymbol{p}_k\|^2$ (memakai $\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k = \alpha_k \boldsymbol{p}_k$). Dengan menggabungkan keduanya, kita peroleh

$$\begin{aligned} & L\alpha_k \|\boldsymbol{p}_k\|^2 \geq (c_2 - 1)\, \nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k \\ \Longrightarrow \quad & \alpha_k \geq \frac{(c_2 - 1)\, \nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k}{L\|\boldsymbol{p}_k\|^2} = \frac{(1 - c_2)\left(-\nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k\right)}{L\|\boldsymbol{p}_k\|^2}, \end{aligned} \tag{B.1}$$

sebab $\boldsymbol{p}_k$ arah turun sehingga $\nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k < 0$. Substitusikan batas bawah α_k pada (B.1) ke syarat Armijo (16.1):

$$\begin{aligned} S(\boldsymbol{\theta}_{k+1}) &\leq S(\boldsymbol{\theta}_k) + c_1 \alpha_k \nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k \\ &\leq S(\boldsymbol{\theta}_k) - \frac{c_1(1 - c_2)}{L} \frac{\left(\nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k\right)^2}{\|\boldsymbol{p}_k\|^2}. \end{aligned}$$

Ingat definisi $\cos\psi_k$ pada (16.3): $\left(\nabla S(\boldsymbol{\theta}_k)^\top \boldsymbol{p}_k\right)^2 / \|\boldsymbol{p}_k\|^2 = \cos^2\psi_k \, \|\nabla S(\boldsymbol{\theta}_k)\|^2$. Maka

$$S(\boldsymbol{\theta}_{k+1}) \le S(\boldsymbol{\theta}_k) - c\,\cos^2\psi_k \, \|\nabla S(\boldsymbol{\theta}_k)\|^2, \qquad c := \frac{c_1(1-c_2)}{L} > 0.$$

Jika pertidaksamaan ini dijumlahkan dari $k = 0$ hingga K, suku-suku di tengah saling meniadakan secara teleskopik:

$$S(\boldsymbol{\theta}_{K+1}) \le S(\boldsymbol{\theta}_0) - c \sum_{k=0}^{K} \cos^2\psi_k \, \|\nabla S(\boldsymbol{\theta}_k)\|^2$$

$$\implies \quad c \sum_{k=0}^{K} \cos^2\psi_k \, \|\nabla S(\boldsymbol{\theta}_k)\|^2 \le S(\boldsymbol{\theta}_0) - S(\boldsymbol{\theta}_{K+1}). \tag{B.2}$$

Karena S terbatas di bawah, ruas kanan terbatas seragam terhadap K (oleh $S(\boldsymbol{\theta}_0) - \inf S$). Dengan mengambil limit $K \to \infty$ pada (B.2), kita memperoleh tepat (16.3). □

Lampiran C

Bukti Teorema Konvergensi Global Levenberg–Marquardt

Lampiran ini memuat bukti Teorema 2 (Subbab 7.2), yang pada badan teks hanya dinyatakan tanpa bukti agar alur baca utama tetap ringan.

Bukti Teorema 2. Andaikan sebaliknya: $\|\nabla S(\boldsymbol{\theta}_k)\| \geq \varepsilon > 0$ untuk semua k (tidak ada barisan bagian menuju titik stasioner). Kita akan menunjukkan bahwa pengandaian ini mengharuskan $S(\boldsymbol{\theta}_k) \to -\infty$, bertentangan dengan asumsi S terbatas di bawah.

Langkah 1 (Setiap langkah yang diterima memiliki penurunan terukur). Dari konstruksi algoritma (baik strategi Marquardt maupun Nielsen), $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{p}_k$ hanya diperbarui ketika $S(\boldsymbol{\theta}_k + \boldsymbol{p}_k) < S(\boldsymbol{\theta}_k)$. Inilah persis kriteria penerimaan. Dari Proposisi 3, begitu $\lambda_k \geq \bar{\lambda}_k$ (nilai ambang yang bergantung pada $\boldsymbol{\theta}_k$), langkah pasti diterima dengan

$$S(\boldsymbol{\theta}_k) - S(\boldsymbol{\theta}_{k+1}) \geq \frac{1}{2\lambda_k}\|\nabla S(\boldsymbol{\theta}_k)\|^2 \geq \frac{\varepsilon^2}{2\lambda_k}. \tag{C.1}$$

Langkah 2 (λ_k terbatas seragam pada saat penerimaan). Karena $J_k^\top J_k$ terbatas seragam pada himpunan level yang kompak, ambang penerimaan pada Proposisi 3 juga dapat dibatasi oleh suatu $\Lambda < \infty$. Strategi Marquardt dan Nielsen memperbesar λ_k setiap kali percobaan gagal. Karena itu, setelah sejumlah percobaan yang berhingga, λ_k melampaui ambang tersebut dan langkah diterima. Nilai λ_k pada saat penerimaan dapat dibatasi oleh $\bar{\Lambda}$.

Langkah 3 (Jumlah teleskopik). Dengan menggabungkan (C.1) dengan batas seragam $\lambda_k \le \bar{\Lambda}$ dari Langkah 2, diperoleh

$$S(\boldsymbol{\theta}_k) - S(\boldsymbol{\theta}_{k+1}) \ge \frac{\varepsilon^2}{2\bar{\Lambda}} =: c > 0 \qquad \text{untuk setiap } k.$$

Dengan menjumlahkan dari $k = 0$ hingga K, kita peroleh

$$S(\boldsymbol{\theta}_0) - S(\boldsymbol{\theta}_{K+1}) \ge (K+1)\,c \quad \Longrightarrow \quad S(\boldsymbol{\theta}_{K+1}) \le S(\boldsymbol{\theta}_0) - (K+1)c.$$

Ruas kanan menuju $-\infty$ ketika $K \to \infty$, bertentangan dengan asumsi S terbatas di bawah. Kontradiksi ini menunjukkan pengandaian di awal salah, sehingga $\|\nabla S(\boldsymbol{\theta}_k)\| \ge \varepsilon$ tidak mungkin berlaku untuk semua k dengan $\varepsilon > 0$ tetap. Artinya, $\liminf_{k\to\infty} \|\nabla S(\boldsymbol{\theta}_k)\| = 0$. □

Lampiran D

Glosarium

Daftar istilah berikut merangkum notasi dan istilah penting yang dipakai berulang kali di seluruh buku. Istilah diurutkan secara alfabetis agar kita dapat menengok ulang suatu istilah tanpa mencari ke bagian aslinya.

Arah turun (*descent direction*)
: Arah $\boldsymbol{p}$ yang membuat fungsi objektif berkurang jika kita melangkah sedikit ke arah tersebut, yaitu $\nabla S^{\top}\boldsymbol{p} < 0$. Lihat Bab 5.

Bilangan kondisi (κ_2)
: Rasio nilai singular terbesar terhadap nilai singular terkecil suatu matriks. Bilangan ini mengukur kepekaan solusi sistem linear terhadap galat pembulatan. Lihat Subbab 4.1.

Daerah kepercayaan (*trust region*)
: Daerah di sekitar iterasi saat ini tempat aproksimasi linear atau kuadrat dari model dapat dipercaya. Radiusnya dilambangkan dengan Δ_k. Lihat Bab 7.

Definit positif
: Sifat matriks simetris A yang memenuhi $\boldsymbol{x}^{\top}A\boldsymbol{x} > 0$ untuk semua $\boldsymbol{x} \neq \boldsymbol{0}$. Sifat ini menjamin A invertibel. Lihat Subbab 3.2.

Entropi-silang (*cross-entropy*)
: Fungsi kerugian untuk klasifikasi yang setara dengan log-kemungkinan negatif model peluang. Lihat Bab 10.

Gradien (∇)
: Vektor yang memuat seluruh turunan parsial pertama suatu fungsi. Gradien menunjuk ke arah kenaikan tercuram fungsi tersebut. Lihat Bab 2.

Hessian (∇^2 atau H)
: Matriks yang memuat seluruh turunan parsial kedua suatu fungsi dan mengukur kelengkungannya. Lihat Bab 3.

IRLS
: *Iteratively Reweighted Least Squares*, yaitu metode Newton untuk regresi logistik dan softmax. Pada setiap iterasi, metode ini menyelesaikan satu masalah kuadrat terkecil berbobot dengan bobot yang diperbarui. Lihat Subbab 10.4.

Jacobian (J)
: Matriks yang memuat seluruh turunan parsial pertama dari fungsi bernilai vektor, di sini residu atau keluaran model, terhadap parameter. Lihat Bab 2.

Konvergensi kuadratik
: Laju konvergensi dengan galat pada iterasi berikutnya sebanding dengan kuadrat galat saat ini ($\|\boldsymbol{e}_{k+1}\| \le C\|\boldsymbol{e}_k\|^2$). Jumlah digit yang akurat kira-kira berlipat ganda pada setiap iterasi. Lihat Bab 6.

Konvergensi linear
: Laju konvergensi dengan galat yang berkurang menurut rasio tetap pada setiap iterasi ($\|\boldsymbol{e}_{k+1}\| \approx \kappa\|\boldsymbol{e}_k\|$). Konvergensi ini lebih lambat daripada konvergensi kuadratik. Lihat Bab 6.

Levenberg–Marquardt
: Perbaikan Gauss–Newton yang menambahkan suku redaman λI pada persamaan normal agar langkah tetap terkendali meskipun $J^\top J$ mendekati singular. Lihat Bab 7.

Matriks Gram
: Matriks yang entrinya berupa hasil kali dalam antarpasangan kolom suatu matriks, misalnya $J^\top J$. Matriks Gram selalu simetris dan semidefinit positif. Lihat Subbab 3.2.

Pencarian garis (*line search*)
: Prosedur memilih panjang langkah α_k pada arah yang tetap, misalnya dengan syarat Armijo dan Wolfe. Lihat Bab 16.

Peringkat penuh kolom
: Sifat matriks J yang kolom-kolomnya bebas linear. Sifat ini setara dengan

$\ker(J) = \{\mathbf{0}\}$ dan menjamin $J^\top J$ definit positif.

Persamaan normal
: Sistem $J^\top J\,\boldsymbol{p} = -J^\top \boldsymbol{r}$ yang diselesaikan pada setiap iterasi Gauss–Newton untuk memperoleh arah langkah. Lihat Subbab 3.3.

Propagasi balik (*backpropagation*)
: Penerapan aturan rantai secara sistematis dari lapisan keluaran menuju lapisan masukan untuk menghitung turunan pada jaringan saraf. Lihat Bab 13 dan 14.

Proyeksi ortogonal
: Titik pada suatu ruang, misalnya ruang kolom J, yang paling dekat dengan sebuah vektor di luar ruang tersebut. Sisa proyeksinya tegak lurus terhadap ruang tersebut. Lihat Subbab 3.3.

Residu (r_i)
: Selisih antara data pengamatan dan prediksi model, yaitu $r_i = y_i - f(\boldsymbol{x}_i, \boldsymbol{\theta})$. Lihat Bab 1.

Ruang kolom (Col**)**
: Himpunan seluruh kombinasi linear dari kolom-kolom suatu matriks.

Semidefinit positif
: Sifat matriks simetris A yang memenuhi $\boldsymbol{x}^\top A\boldsymbol{x} \geq 0$ untuk semua $\boldsymbol{x}$. Sifat ini lebih lemah daripada definit positif.

SVD (dekomposisi nilai singular)
: Faktorisasi $J = U\Sigma V^\top$ yang mengungkap nilai singular, yaitu faktor peregangan suatu matriks pada arah-arah utamanya. SVD menjadi dasar analisis bilangan kondisi dan penyelesaian kasus kekurangan peringkat. Lihat Subbab 4.1.

Daftar Pustaka

S.-I. Amari. "Natural Gradient Works Efficiently in Learning." *Neural Computation*, 10(2):251–276, 1998. DOI: https://doi.org/10.1162/089976698300017746.

L. Armijo. "Minimization of Functions Having Lipschitz Continuous First Partial Derivatives." *Pacific Journal of Mathematics*, 16(1):1–3, 1966.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

Å. Björck. *Numerical Methods for Least Squares Problems*. SIAM, 1996. DOI: https://doi.org/10.1137/1.9781611971484.

J. E. Dennis, Jr. dan R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Classics in Applied Mathematics 16. SIAM, 1996.

R. A. Fisher. "Iris." UCI Machine Learning Repository. 1936. DOI: https://doi.org/10.24432/C56C76.

G. H. Golub dan C. F. Van Loan. *Matrix Computations*, edisi keempat. Johns Hopkins University Press, 2013.

P. J. Green. "Iteratively Reweighted Least Squares for Maximum Likelihood Estimation, and Some Robust and Resistant Alternatives." *Journal of the Royal Statistical Society: Series B*, 46(2):149–192, 1984.

M. T. Hagan dan M. B. Menhaj. “Training Feedforward Networks with the Marquardt Algorithm.” *IEEE Transactions on Neural Networks*, 5(6):989–993, 1994. DOI: https://doi.org/10.1109/72.329697.

Kaggle. “Titanic: Machine Learning from Disaster.” https://www.kaggle.com/competitions/titanic/data. Diakses 18 Juli 2026.

K. Levenberg. “A Method for the Solution of Certain Non-Linear Problems in Least Squares.” *Quarterly of Applied Mathematics*, 2(2):164–168, 1944.

K. Madsen, H. B. Nielsen, dan O. Tingleff. *Methods for Non-Linear Least Squares Problems*, edisi kedua. Technical University of Denmark, 2004. https://www2.imm.dtu.dk/pubdb/pubs/3215-full.html.

D. W. Marquardt. “An Algorithm for Least-Squares Estimation of Nonlinear Parameters.” *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

J. Martens. “New Insights and Perspectives on the Natural Gradient Method.” *Journal of Machine Learning Research*, 21(146):1–76, 2020.

P. McCullagh dan J. A. Nelder. *Generalized Linear Models*, edisi kedua. Chapman and Hall, 1989.

J. J. Moré. “The Levenberg–Marquardt Algorithm: Implementation and Theory.” Dalam G. A. Watson (Ed.), *Numerical Analysis*, Lecture Notes in Mathematics 630, hlm. 105–116. Springer, 1978.

J. J. Moré dan D. C. Sorensen. “Computing a Trust Region Step.” *SIAM Journal on Scientific and Statistical Computing*, 4(3):553–572, 1983.

J. A. Nelder dan R. W. M. Wedderburn. “Generalized Linear Models.” *Journal of the Royal Statistical Society: Series A*, 135(3):370–384, 1972.

H. B. Nielsen. *Damping Parameter in Marquardt’s Method*. Technical Report IMM-REP-1999-05, Technical University of Denmark, 1999. https://www.imm.dtu.dk/documents/ftp/tr99/tr05_99.abstract.html.

J. Nocedal dan S. J. Wright. *Numerical Optimization*, edisi kedua. Springer, 2006. https://link.springer.com/book/10.1007/978-0-387-40065-5.

M. J. D. Powell. "A Hybrid Method for Nonlinear Equations." Dalam P. Rabinowitz (Ed.), *Numerical Methods for Nonlinear Algebraic Equations*, hlm. 87–114. Gordon and Breach, 1970.

D. E. Rumelhart, G. E. Hinton, dan R. J. Williams. "Learning Representations by Back-Propagating Errors." *Nature*, 323(6088):533–536, 1986. DOI: https://doi.org/10.1038/323533a0.

N. N. Schraudolph. "Fast Curvature Matrix-Vector Products for Second-Order Gradient Descent." *Neural Computation*, 14(7):1723–1738, 2002.

Scikit-learn Developers. "Diabetes Dataset." https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_diabetes.html. Diakses 18 Juli 2026.

L. N. Trefethen dan D. Bau. *Numerical Linear Algebra*. SIAM, 1997.

W. Wolberg, O. Mangasarian, N. Street, dan W. Street. "Breast Cancer Wisconsin (Diagnostic)." UCI Machine Learning Repository. 1993. DOI: https://doi.org/10.24432/C5DW2B.

P. Wolfe. "Convergence Conditions for Ascent Methods." *SIAM Review*, 11(2):226–235, 1969.

G. Zoutendijk. "Nonlinear Programming, Computational Methods." Dalam J. Abadie (Ed.), *Integer and Nonlinear Programming*, hlm. 37–86. North-Holland, 1970.

Tentang Penulis

Dr. Mohammad Jamhuri, M.Si adalah dosen pada Program Studi Matematika, Fakultas Sains dan Teknologi, UIN Maulana Malik Ibrahim Malang. Bidang keahliannya adalah analisis numerik dan pembelajaran mesin, terutama metode optimisasi berbasis turunan kedua. Surel: m.jamhuri@mat.uin-malang.ac.id.

Prof. Dr. Sri Harini, M.Si adalah guru besar statistika pada Program Studi Matematika, Fakultas Sains dan Teknologi, UIN Maulana Malik Ibrahim Malang. Bidang keahliannya adalah statistika dan analisis deret runtun waktu. Surel: sriharini@mat.uin-malang.ac.id.

Andy Irawan, M.Kom berkarya di Fakultas Sains dan Teknologi, UIN Maulana Malik Ibrahim Malang. Minat penelitiannya meliputi informatika dan matematika, terutama jaringan saraf dalam yang dilatih dengan optimisasi bebas-Hessian.

Juhari, M.Si adalah dosen matematika terapan pada Program Studi Matematika, Fakultas Sains dan Teknologi, UIN Maulana Malik Ibrahim Malang. Penelitiannya berfokus pada analisis dinamik model matematika. Surel: juhari@uin-malang.ac.id.

Mutiara Alvi, S.Si dan **Natasyah Asviana, S.Si** adalah lulusan sarjana Program Studi Matematika, Fakultas Sains dan Teknologi, UIN Maulana Malik Ibrahim Malang.