FUDAN UNIVERSITY

DRAFT VERSION 2.0
2026 LECTURE NOTES

Machine Learning Algorithms for Data Science and Physics Applications

An Elementary Introduction

Bao-Jun Cai *

Institute of Modern Physics, Fudan University

Yu-Gang Ma †

Institute of Modern Physics, Fudan University
College of Physics, East China Normal University

7/3/2026‡§

*bjcai@fudan.edu.cn
†mayugang@fudan.edu.cn

‡starting from 7/5/2025
§Comments are welcome!

Preface

These lecture notes were prepared for the course *Introduction to Algorithms with Applications in Data Science and Physics Problems*, taught at the Institute of Modern Physics (IMP) of Fudan University in 2026. Much of the material originated from a series of lectures previously delivered at Shadow Creator during 2018-2022, as well as from earlier courses entitled *Introduction to Algorithms for Machine Learning* and *State Estimation*. The notes have since been substantially reorganized and expanded for university-level teaching.

The material is organized into a sequence of lectures, each devoted to a particular topic and designed to be completed within a single class session of approximately 2-3 hours. Consequently, each chapter is largely self-contained and contains sufficient material for one lecture. Since the course is intended for students from diverse academic backgrounds, particular emphasis is placed on fundamental concepts and broadly applicable techniques. The central philosophy of these notes is that many important problems arising in data science, machine learning, scientific computing, computer vision, and the physical sciences can be understood and solved using a relatively small collection of mathematical and algorithmic principles. Our hope is that these notes will provide readers with useful computational tools and algorithmic insights applicable to a wide variety of scientific and engineering problems. More specifically,

(a) Lectures 1-3 provide a concise review of calculus together with the foundations of probability and statistics, emphasizing both theoretical concepts and numerical implementations.

(b) Lectures 4-8 introduce several widely used optimization algorithms, including both first-order and second-order methods.

(c) Lectures 9 and 10 discuss two representative state-estimation problems: the Kalman filter and stereo reconstruction in computer vision.

(d) Lecture 11 presents an introduction to Bayesian inference, using linear curve fitting as a concrete example.

(e) Lectures 12-15 focus primarily on high-dimensional problems, where principal component analysis, singular-value decomposition, and random walks play central roles. Robust estimation techniques and randomized algorithms for matrix multiplication are also discussed.

(f) Lectures 16-19 introduce basic concepts from graph theory, ranging from random graphs to regular graphs and their applications.

(g) Lecture 20 is devoted to introductory algorithms for ordinary and partial differential equations, illustrating how ideas such as random walks can also be applied in numerical analysis.

(h) Lecture 21 introduces the fundamental ideas of the Fast Fourier Transform (FFT), one of the most important computational tools in modern scientific computing and data analysis.

(i) Lectures 22-24 provide a brief introduction to quantum mechanics and several elementary quantum algorithms. Particular emphasis is placed on the path-integral formulation of quantum mechanics and on the essential role played by the imaginary unit $\sqrt{-1}$ in quantum interference phenomena.

(j) An appendix is included to introduce more applications of perturbation methods.

Each lecture contains a number of exercises integrated into the main discussion, together with additional problems at the end of the lecture for further study. The placement of the exercises follows the style adopted by Landau and Lifshitz in their *Course of Theoretical Physics* and by Thorne and Blandford in *Modern Classical Physics*, where problems are interwoven with the exposition rather than being collected separately. The problems at the end of each lecture are intended to provide more extensive practice and to encourage further exploration of the subject. We strongly encourage readers to work through these exercises, particularly the computational and programming problems, since practical implementation is often the most effective way to master new techniques. *There is* *little doubt that problem-solving constitutes one of the most important aspects of learning any challenging subject.* It is therefore our hope that the exercises provided throughout these notes will help readers develop both a deeper understanding of the underlying concepts and greater confidence in applying them. Many of the problems are designed not only to reinforce theoretical ideas but also to cultivate practical computational skills.* In the original versions of these courses, each assignment typically included one or two computational projects requiring students to derive relevant formulas, implement algorithms, and analyze the resulting data using programming languages of their choice, such as Python, C++, Java, Julia, or Fortran. Readers are encouraged to adopt a similar approach when studying these notes. A basic familiarity with calculus, probability theory, and linear algebra is assumed throughout; nevertheless, whenever additional background knowledge is required, the necessary concepts are introduced briefly so that the presentation remains as self-contained as possible.

The primary aim of this volume is to introduce the fundamental ideas underlying modern data science, scientific computing, and machine learning rather than to provide a comprehensive survey of contemporary algorithms. To this end, we focus on essential topics such as probability and statistics, numerical computation, optimization, state estimation, high-dimensional geometry, graph theory, Fourier analysis, and related algorithmic foundations. These subjects constitute the common language of modern computational science and underpin a wide range of methods in data science, physics, engineering, and beyond. Although machine learning and artificial intelligence continue to evolve rapidly, their mathematical foundations remain remarkably stable. Many seemingly different algorithms can be understood as variations or extensions of a relatively small set of core ideas, including optimization, statistical inference, dimensionality reduction, spectral analysis, stochastic processes, and numerical approximation. A solid understanding of these principles not only facilitates the study of existing methods but also provides the tools needed to adapt to future developments. For this reason, we adopt a deliberately conservative approach, emphasizing concepts that are likely to remain important regardless of technological change. In our experience, a deep understanding of fundamental principles is far more valuable than a superficial familiarity with numerous sophisticated algorithms. Advanced topics such as deep neural networks, transformers, large language models, diffusion models, reinforcement learning, generative artificial intelligence, probabilistic graphical models, and large-scale distributed learning systems are therefore omitted or discussed only briefly. *The present volume is intended as an introduction to the algorithmic foundations of data science and computational science. A future companion volume will explore a number of modern machine-learning and artificial-intelligence methods in greater depth.*

We provide several possible shorter courses based on selected chapters from these notes. The suggested number of class sessions for each course is indicated below:

(a) *Foundations of Numerical Computing*: Lectures 1, 2, 3, 6, and 20. These lectures provide an elementary introduction to numerical calculus, Monte Carlo methods, and finite-difference algorithms for ordinary and partial differential equations. Particular emphasis is placed on approximation techniques, numerical integration, random sampling, and the basic philosophy of scientific computing. (6-8)

(b) *Methods of Optimization*: Lectures 4, 5, 7, 8, 11, 13, and 14. This sequence introduces first- and second-order optimization methods, including gradient descent, conjugate-gradient, and Newton algorithms, together with basic ideas of Bayesian inference. The optimization perspectives underlying principal component analysis and singular-value decomposition are also discussed. (8-10)

(c) *State Estimation*: Lectures 9, 10, and 11. These lectures provide a compact introduction to state estimation through Kalman filtering and Bayesian inference, with applications to computer vision problems such as bundle adjustment, stereo depth estimation, and simultaneous localiza-

*For example, see the discussion by Academician Xi-Ru Chen in the preface to his textbook *Advanced Statistics*, where the importance of problem-solving is particularly emphasized.

tion and mapping (SLAM). (4-6)

(d) *High-dimensional Problems and Randomized Algorithms*: Lectures 6, 12, 13, 14, 15, 16, 17, and 18. Beginning with random walks, this sequence develops the concepts of randomness, high-dimensional geometry, graph structures, and fast randomized algorithms that trade a controlled loss of accuracy for significant computational gains. (10-12)

(e) *Graph Theory and Network Science*: Lectures 16, 17, 18, and 19. Topics include graph representations, random graphs, random walks on networks, graph connectivity, network dynamics, and matrix-factorization methods. Applications to social networks, recommendation systems, and large-scale data analysis may also be discussed. (6-8)

(f) *Algorithms for Scientific Computing in Physics*: Lectures 2, 3, 6, 20, and 21. These lectures focus on computational techniques frequently encountered in scientific research, including numerical integration, Monte Carlo methods, differential-equation solvers, and spectral methods based on the Fast Fourier Transform. Representative examples from physics and engineering can be incorporated throughout the course. (8-10)

(g) *Signal Processing*: Lectures 21, 3, 13, and 14. This course introduces the fundamental concepts of signal representation, frequency-domain analysis, dimensionality reduction, and spectral decomposition. The Fast Fourier Transform is emphasized as one of the most influential algorithms in modern computational science. (5-7)

(h) *Introduction to Quantum Algorithms*: Lectures 22-24, together with selected material from Lectures 13 and 14. This course introduces the basic principles of quantum mechanics relevant to computation, including quantum states, interference, path integrals, and elementary quantum algorithms. Particular attention is devoted to understanding the algorithmic role of complex amplitudes and quantum interference. (4-6)

These notes are not intended to provide a comprehensive treatment of any single topic. Rather, they aim to offer a broad introduction to a collection of important ideas and techniques that frequently arise in modern data science, machine learning, scientific computing, computer vision, and the physical sciences. If they succeed in helping readers acquire both theoretical understanding and practical computational skills, while at the same time cultivating the intuition needed to approach future developments in machine learning, scientific computing, and artificial intelligence, then the primary goal of this work will have been achieved.

*****Acknowledgments*****

Partial Book List

The preparation of these lecture notes has benefited greatly from numerous excellent textbooks and monographs. A selection of particularly useful references is listed below:†

1. *S. Aaronson, *Quantum Computing Since Democritus*, Cambridge University Press, 2013. [*an engaging introduction to quantum computing, computational complexity, and the conceptual foundations of information and physics*]
2. *A. Barabási, *Network Science*, Cambridge University Press, 2016. [*an accessible introduction to complex networks, emphasizing scale-free networks, network dynamics, and data-driven applications*]
3. D. Barber, *Bayesian Reasoning and Machine Learning*, Cambridge University Press, 2012.
4. T. Barfoot, *State Estimation for Robotics*, Cambridge University Press, 2017.
5. *G. Benenti, G. Casati, D. Rossini, and G. Strini, *Principles of Quantum Computation and Information: A Comprehensive Textbook*, World Scientific, 2019. [*a comprehensive and self-contained introduction to quantum computation and quantum information, covering quantum mechanics, quantum circuits, quantum algorithms, quantum cryptography, error correction, and quantum simulation from both physical and computational perspectives*]
6. *D. Bertsekas, *Dynamic Programming and Optimal Control*, Athena Scientific, 2017. [*the standard reference on dynamic programming, optimal control, and reinforcement learning foundations*]
7. *C. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006. [*excellent introduction to probabilistic inference and machine learning from a Bayesian perspective*]
8. C. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.
9. C. Bishop and H. Bishop, *Deep Learning: Foundations and Concepts*, Springer, 2024.
10. *A. Blum, J. Hopcroft, and R. Kannan, *Foundations of Data Science*, Cambridge University Press, 2020. [*accessible introduction to modern high-dimensional methods and their applications in data science*]
11. L. Böttcher and H. Herrmann, *Computational Statistical Physics*, Cambridge University Press, 2021.
12. S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
13. S. Brunton and J. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*, Cambridge University Press, 2022.
14. G. Casella and R. Berger, *Statistical Inference*, Thompson, 2006.
15. **T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, 2022. [*the standard reference on algorithms and data structures*]
16. T. Cover and J. Thomas, *Elements of Information Theory*, Wiley, 2006.
17. G. Cowan, *Statistical Data Analysis*, Oxford University Press, 1998.
18. S. Dasgupta, C. Papadimitriou, and U. Vazirani, *Algorithms*, McGraw-Hill Education, 2006.
19. *M. Deisenroth, A. Faisal, and C. Ong, *Mathematics for Machine Learning*, Cambridge University Press, 2020. [*a modern introduction to the mathematical foundations of machine learning, emphasizing linear algebra, optimization, probability, and statistical modeling*]
20. S. Dorogovtsev and J. Mendes, *The Nature of Complex Networks*, Oxford University Press, 2025.
21. *R. Durrett, *Probability: Theory and Examples*, Cambridge University Press, 2019. [*a modern and rigorous introduction to probability theory*]
22. J.Q. Fan, R.Z. Li, C.H. Zhang, and H. Zou, *Statistical Foundations of Data Science*, CRC Press, 2020.
23. A. Gelman, J. Carlin, H. Stern, D. Dunson, A. Vehtari, and D. Rubin, *Bayesian Data Analysis*, CRC Press, 2013.
24. **G. Golub and Van C. Loan, *Matrix Computations*, Johns Hopkins University Press, 2013. [*the classic reference on numerical linear algebra and matrix algorithms*]
25. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

†Books marked with "*" are especially recommended for readers interested in pursuing the subject in greater depth. The books listed provide useful background material and complementary perspectives on the subjects covered in these lectures. They are not intended to constitute a complete bibliography. Appropriate references to original research papers and review articles will be given throughout the text as specific topics arise. Readers wishing to pursue a subject in greater depth are encouraged to consult both the references given here and those cited in the individual lectures.

26. M. Harchol-Balter, *Introduction to Probability for Computing*, Cambridge University Press, 2024.
27. R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2004.
28. *T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, 2008. [*a comprehensive treatment of classical statistical learning methods*]
29. R. Horn and C. Johnson, *Matrix Analysis*, Cambridge University Press, 2012.
30. A. Izenman, *Modern Multivariate Statistical Techniques*, Springer, 2008.
31. I. Jolliffe, *Principal Component Analysis*, Springer, 1986.
32. *M. Kalos and P. Whitlock, *Monte Carlo Methods*, Wiley, 1986. [*clear and insightful introduction to classical and quantum Monte Carlo techniques*]
33. J. Kleinberg and E. Tardos, *Algorithm Design*, Pearson, 2005.
34. M. Kochenderfer and T. Wheeler, *Algorithms for Optimization*, MIT Press, 2019.
35. D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*, MIT Press, 2009.
36. W. Krauth, *Statistical Physics: Algorithms and Computations*, Oxford University Press, 2006.
37. * D. Landau and K. Binder, *A Guide to Monte Carlo Simulations in Statistical Physics*, Cambridge University Press, 2021. [*one of the most comprehensive introductions to Monte Carlo in statistical physics*]
38. E. Lehmann and G. Casella, *Theory of Point Estimation*, Springer, 1998.
39. R. LeVeque, *Finite Difference Methods for Ordinary and Partial Differential Equations*, SIAM, 2007.
40. *J. Liu, *Monte Carlo Strategies in Scientific Computing*, Springer, 2008. [*bridges Monte Carlo methods, Bayesian computation, and scientific computing*]
41. **D. MacKay, *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press, 2003. [*an outstanding introduction to information theory, Bayesian inference, and machine learning from a physicist's perspective*]
42. *N. Mermin, *Quantum Computer Science*, Cambridge University Press, 2007. [*emphasizes algorithmic thinking rather than hardware implementation*]
43. M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*, Cambridge University Press, 2017.
44. M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*, MIT Press, 2018.
45. Y. Mostafa, M. Ismail, and H.T. Lin, *Learning from Data*, AMLBook, 2012.
46. R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.
47. K. Murphy, *Machine Learning: A Probabilistic Perspective*, MIT Press, 2012.
48. R. Neal, *Bayesian Learning for Neural Networks*, Springer, 1996.
49. Y. Nesterov, *Lectures on Convex Optimization*, Springer, 2018.
50. *M. Newman, *Networks*, Oxford University Press, 2018. [*the standard textbook on network science and complex networks*]
51. M. Newman and G. Barkema, *Monte Carlo Methods in Statistical Physics*, Clarendon Press, 1999.
52. *M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, 2010. [*the definitive introduction to quantum computation and quantum information*]
53. J. Nocedal and S. Wright, *Numerical Optimization*, Springer, 2006.
54. P. Norvig and S. Russell, *Artificial Intelligence: A Modern Approach*, Pearson, 2020.
55. *J. Pearl, *Causality*, Cambridge University Press, 2009. [*the foundational reference on causal inference and graphical models*]
56. **W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C++*, Cambridge University Press, 2007. [*a broad survey of numerical algorithms for scientific computing and data analysis*]
57. S. Prince, *Computer Vision: Models, Learning, and Inference*, Cambridge University Press, 2012.
58. *S. Prince, *Understanding Deep Learning*, MIT Press, 2023. [*one of the clearest modern introductions to deep learning, emphasizing intuition, mathematics, and implementation*]
59. C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*, MIT Press, 2005.
60. C. Robert and G. Casella, *Monte Carlo Statistical Methods*, Springer, 2004.
61. B. Schölkopf and A. Smola, *Learning with Kernels*, Cambridge University Press, 2002.
62. S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*, Cambridge University Press, 2014.

63. D. Sivia and J. Skilling, *Data Analysis: A Bayesian Tutorial*, Oxford University Press, 2006.
64. S. Skiena, *The Algorithm Design Manual*, Springer, 2020.
65. *G. Strang, *Introduction to Linear Algebra*, Wellesley-Cambridge Press, 2023. [*arguably the most popular introduction to applied linear algebra*]
66. *R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 2018. [*the definitive introduction to reinforcement learning and sequential decision making*]
67. R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer, 2022.
68. S. Theodoridis, *Machine Learning: A Bayesian and Optimization Perspective*, Academic Press, 2020.
69. *A. Torralba, P. Isola, and W. Freeman, *Foundations of Computer Vision*, MIT Press, 2024. [*a modern and comprehensive introduction to computer vision, unifying classical geometric methods with contemporary deep-learning approaches*]
70. L. Trefethen and D. Bau, *Numerical Linear Algebra*, SIAM, 1997.
71. S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*, MIT Press, 2005.
72. R. Vershynin, *High-dimensional Probability: An Introduction with Applications in Data Science*, Cambridge University Press, 2018.
73. R. Vidal, Y. Ma, and S. Sastry, *Generalized Principal Component Analysis*, Springer, 2016.
74. *M. Wainwright, *High-dimensional Statistics: A Non-asymptotic Viewpoint*, Cambridge University Press, 2018. [*an excellent introduction to modern high-dimensional probability and statistical theory*]
75. *J. Watrous, *The Theory of Quantum Information*, Cambridge University Press, 2018. [*a rigorous treatment of the mathematical foundations underlying quantum algorithms*]
76. *M. Wilde, *Quantum Information Theory*, Cambridge University Press, 2017. [*a modern graduate-level treatment of quantum information, quantum communication, and information-processing protocols*]
77. J. Wright and Y. Ma, *High-dimensional Data Analysis with Low-dimensional Models: Principles, Computation, and Applications*, Cambridge University Press, 2022.

— Brief Contents —

— Detailed Contents —

Machine Learning Algorithms for Data Science and Physics Applications

Lecture 1 Order of Magnitude and Divide-and-Conquer

Key Concepts/Topics of This Lecture
guess and estimate, order of magnitude
period of a simple pendulum, perturbative calculations
master theorem, divide-and-conquer, Fibonacci series
fast algorithm for matrix multiplication
algebraic equation $x^n(t)=\Omega+t\,x(t)/\Lambda$, $x(t)\in\mathrm{R}^+$, $n\geq 5$, $n\in\mathrm{N}^+$

§1.1 General Idea

In situations where one attempts to calculate and/or estimate the value of a quantity W without any prior information, the first essential step is to determine its sign, i.e., whether it is positive or negative. Moreover, one should roughly estimate its order of magnitude, for example, whether $|W|$ is of order 1, 10, or 100, etc. Once this is clarified, one gains a preliminary understanding of the general behavior of W. If the exact form of W can be written as

$$\boxed{W=\omega\times 10^{\delta},} \tag{1.1}$$

then the purpose of an estimate is to roughly determine the index δ, while an accurate evaluation of ω typically requires a complete theoretical framework.

"Exploration by guessing" is an important and direct approach. However, it should always be guided by the underlying principles of the problem under consideration. Another useful tool for estimating a quantity is perturbative calculation. The fundamental idea of perturbative calculations is to identify a suitably small quantity (the perturbative element) and then carry out the calculation order by order, usually through Taylor expansion. For example, if a quantity f can be obtained perturbatively, one assumes a perturbative expansion of the form $f=f_0+f_1+f_2+\cdots$, where f_0 is called the leading-order term, and provides the dominant contribution to f. On the other hand, $f_1+f_2+\cdots$ collectively represent corrections, which are often small in the sense that ($j=1,2,\cdots$),

$$\boxed{\left|\frac{f_j}{f_0}\right|\ll 1,\ \ \frac{\sum_j|f_j|}{|f_0|}\ll 1,\ \ \left|\frac{f_{j+1}}{f_j}\right|<1,\ \ \lim_{j\to\infty}\frac{f_{j+1}}{f_j}=0.} \tag{1.2}$$

In general, the quantity f depends on several parameters, namely $f=f\{\phi_i\}=f(\phi_1,\cdots,\phi_n)$, among which certain parameters ϕ_α may be appropriate for constructing a perturbative scheme. For example,

$$f\{\phi_i\}=f_{0,\alpha}\left(1+g_{1,\alpha}\phi_\alpha+g_{2,\alpha}\phi_\alpha^2+\cdots\right), \tag{1.3}$$

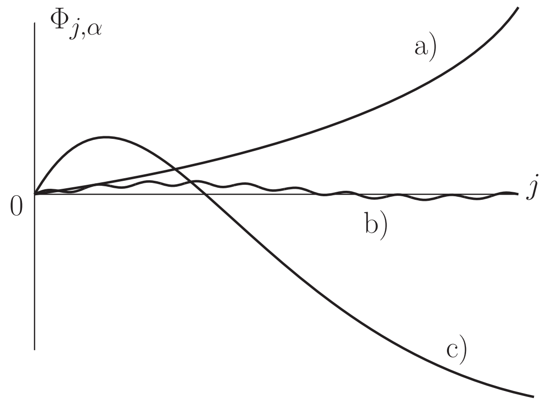

FIG. 1.1: Convergence of the perturbative calculations, where patterns a) and c) are unreasonable.

where $g_{1,\alpha}, g_{2,\alpha}, \cdots$ are the expansion coefficients, and $f_{0,\alpha}$ denotes the corresponding value at $\phi_\alpha = 0$. One may define an effective convergence factor as $\Phi_{j,\alpha} = g_{j+1,\alpha}/g_{j,\alpha}$, which fundamentally satisfies $\lim_{j\to\infty}\Phi_{j,\alpha} = 0$. In many practical problems, the factor $\Phi_{j,\alpha}$ approaches zero rapidly, as illustrated in FIG. 1.1. In this figure, pattern b) represents a reasonable behavior, whereas patterns a) and c) are not suitable for perturbative calculations.

In addition, numerical algorithms and simulations are also essential for certain problems, and these techniques will be further developed in the current note.

§1.2 Period of a Simple Pendulum

From an elementary middle-school physics course, we know that the period of a simple pendulum with a small swing/oscillation angle (e.g., $\lesssim 5^\circ$) is given by

$$\boxed{T = 2\pi\sqrt{\ell/g},} \tag{1.4}$$

where ℓ is the length of the pendulum and g is the gravitational acceleration. For a simple pendulum, the relevant physical quantities are the length (ℓ, in m), the mass of the bob (m, in kg), the gravitational acceleration (g, in m/s^2), and the maximum swing angle ($\chi_{\max}$, dimensionless).

EXERCISE 1-1. Derive (1.4) by solving the equation of motion in the small-angle approximation.

EXERCISE 1-2. If a quantity f has dimension $[\mathrm{M}]^\mu[\mathrm{L}]^\nu[\mathrm{T}]^\sigma$, where M, L, and T denote the basic units of mass, length, and time, can μ, ν, and σ take arbitrary real values?

Since $\chi_{\max}$ is dimensionless and therefore can appear only in purely numerical factors, without carrying any physical dimension. Moreover, as the swing angle increases, the period of the system also increases. For example, when the swing angle reaches π, the period becomes infinite. Based on these observations, $\chi_{\max}$ is expected to enter in the following way:

$$\boxed{T = 2\pi\sqrt{\frac{\ell}{g}} \times \left(1 + a\chi_{\max}^2 + \cdots\right),} \tag{1.5}$$

where the coefficient a is positive.

EXERCISE 1-3. Can one estimate the value of a without doing detailed calculations? Based on which estimate the leading nonlinear effect for, e.g., $\chi_{\max} = \pi/4$.

Clearly, without knowing the value of a, formula (1.5) is not practically useful. We now present the analytical derivation of the period of a simple pendulum for a general swing angle. The equation of motion can be written as

$$\frac{1}{2}mv^2(t) + U(x,t) = E, \quad v(t) = \frac{\mathrm{d}x}{\mathrm{d}t}. \tag{1.6}$$

If the potential is time independent, then $U(x,t) = U(x)$. Solving this equation gives

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \sqrt{\frac{2[E-U(x)]}{m}} \to t = \sqrt{\frac{m}{2}} \int \frac{\mathrm{d}x}{\sqrt{E-U(x)}}. \tag{1.7}$$

The period of the system is therefore [∗1-1∗]

$$T = 4\sqrt{\frac{m}{2}} \int_0^{\chi_{\max}} \frac{\mathrm{d}x}{\sqrt{E-U(x)}} = 4\sqrt{\frac{\ell}{g}} \int_0^{\pi/2} \frac{\mathrm{d}\vartheta}{\sqrt{1-\sin^2\frac{\chi_{\max}}{2}\sin^2\vartheta}}. \tag{1.8}$$

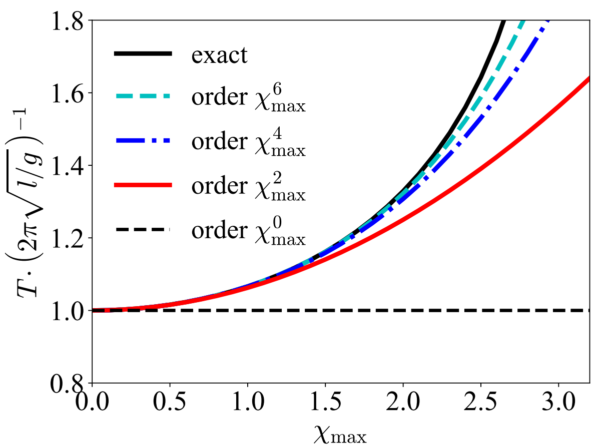

FIG. 1.2: The reduced period $T(\ell/g)^{-1/2}/2\pi$ as a function of $\chi_{\max}$, see formula (1.12).

According to (1.8), in the zeroth-order approximation ($\chi_{\max} \approx 0$, small angle), we have

$$T \approx 4\sqrt{\frac{\ell}{g}} \int_0^{\pi/2} \mathrm{d}\vartheta = 2\pi\sqrt{\frac{\ell}{g}}. \tag{1.9}$$

In the first-order approximation, one expands

$$\left(1-\sin^2\frac{\chi_{\max}}{2}\sin^2\vartheta\right)^{-1/2} \approx 1 + \frac{1}{8}\chi_{\max}^2\sin^2\vartheta, \tag{1.10}$$

which leads to

$$T \approx 2\pi\sqrt{\frac{\ell}{g}} \times \left(1 + \frac{1}{16}\chi_{\max}^2\right). \tag{1.11}$$

For general motion with arbitrary $\chi_{\max}$, straightforward calculation yields

$$T \approx 2\pi\sqrt{\frac{\ell}{g}} \times \left(1 + \frac{1}{16}\chi_{\max}^2 + \frac{11}{3072}\chi_{\max}^4 + \frac{173}{737280}\chi_{\max}^6\right). \tag{1.12}$$

In FIG. 1.2, the reduced period $T(\ell/g)^{-1/2}/2\pi$ is plotted as a function of $\chi_{\max}$ at different orders in $\chi_{\max}$. If $\chi_{\max} = \pi$, the integral in (1.8) diverges, indicating that the perturbative expansion breaks down. From a physical viewpoint, $\chi_{\max} = \pi$ corresponds to an unstable equilibrium configuration, and therefore the system has no finite period (equivalently, the period is infinite).

The coefficients 1/16, 11/3072, and 173/737280 ensure that each successive term is indeed a perturbative correction, even though $\chi_{\max}$ itself can be larger than 1.

EXERCISE 1-4. How can one numerically evaluate the period (1.8) for a general angle $\chi_{\max}$?

EXERCISE 1-5. Derive the next-order coefficient 11/3072 in (1.12) and explain its origin.

§1.3 Solution of an Algebraic Equation

The main feature of order-of-magnitude estimate and approximate perturbative calculations can already be clearly illustrated through very elementary mathematical examples. As a simple case, consider the algebraic equation

$$x^n(t) = \Omega + t\,x(t)/\Lambda, \ \ x(t) \in \mathrm{R}^+, \ \ n \in \mathrm{N}^+, \ \ n \geq 5, \ \ t \geq 0, \tag{1.13}$$

where t and Λ are two parameters (for example, t may represent time and start from zero), and Ω and Λ are positive constants. As is well known, there is no general closed-form solution for algebraic equations of degree higher than four. Therefore, for equations such as (1.13), one must rely either on numerical algorithms or on appropriate approximation methods to determine the root.

If the "time" t is small and close to zero, the second term on the right-hand side of Eq. (1.13) can be treated as a perturbation. In this case, we assume $x(t) \approx x_0[1+\delta(t)] = \Omega^{1/n}[1+\delta(t)]$, where $\delta(t)$ is a small correction to the leading-order solution. Expanding both sides of Eq. (1.13) to first order in $\delta(t)$, one obtains $\delta(t) = t\Omega^{1/n-1}/n\Lambda$, and therefore

$$x(t) \approx \Omega^{1/n}\left(1 + \frac{1}{n\Lambda}\Omega^{1/n-1}t\right). \tag{1.14}$$

The above approximate theory breaks down when $t \lesssim n\Lambda\Omega^{1-1/n}$. The approximation adopted here is usually referred to as linear perturbation theory, and the basic requirement is that t should not be larger than $n\Lambda\Omega^{1-1/n}$, for example $t \lesssim t_{\max} \approx s\,n\Lambda\Omega^{1-1/n}$ with $s \ll 1$.

If one includes the next contribution to the solution by writing $x(t) = \Omega^{1/n}(1 + \alpha t + \beta t^{\sigma})$, where $\alpha = \Omega^{1/n-1}/n\Lambda$, then straightforward calculation shows that $\sigma = 2$, and consequently

$$x(t) \approx \Omega^{1/n}\left(1 + \frac{1}{n\Lambda}\Omega^{1/n-1}t - \frac{n-3}{2n^2\Lambda^2}\Omega^{2/n-2}t^2\right). \tag{1.15}$$

From the first-order analysis we have the condition $t \ll n\Lambda\Omega^{1-1/n}$. However, when higher-order corrections are included, such as the second-order term in (1.15), the perturbative requirement becomes $|(n-3)\delta^2(t)/2| \ll |\delta(t)|$, or equivalently $|(n-3)\delta(t)/2| \ll 1$, which implies

$$t \ll \frac{2n\Lambda\Omega^{1-1/n}}{n-3}. \tag{1.16}$$

If n is very large, this condition reduces to $t \ll 2\Lambda\Omega^{1-1/n} \sim 2\Lambda\Omega$, which is weaker than the original criterion $t \ll n\Lambda\Omega^{1-1/n}$. This indicates that the effective perturbative region shrinks as higher-order terms are taken into account.

The perturbative parameter in Eq. (1.13) is in fact $\delta(t) = \alpha t = \Omega^{1/n-1} t/n\Lambda$, and therefore $x(t) \approx \Omega^{1/n}[1+\delta(t)-(n-3)\delta^2(t)/2]$. It is evident that this quantity can no longer be regarded as small when t becomes large, indicating that both the linear theory and its higher-order extensions eventually break down at sufficiently large t.

On the other hand, in the limit $t \to \infty$, a different perturbative scheme naturally emerges. In this regime, the term Ω on the right-hand side of Eq. (1.13) can be neglected, leading to $x_\infty(t) = (t/\Lambda)^{1/(n-1)}$, which is referred to as the asymptotic (large-t) solution of Eq. (1.13). Assuming $x(t) \approx x_\infty(t)[1+\phi(t)]$ with $|\phi(t)| \ll 1$, one then obtains

$$x(t) \approx \left(\frac{t}{\Lambda}\right)^{\frac{1}{n-1}} \left[1 + \frac{\Omega}{n-1}\left(\frac{\Lambda}{t}\right)^{\frac{n}{n-1}}\right], \tag{1.17}$$

with the condition

$$t \gg t_{\text{asp}} \equiv \Lambda \exp\left(-\frac{n-1}{n}\ln\frac{n-1}{\Omega}\right). \tag{1.18}$$

Moreover, proceeding to even higher order by writing $x(t) \approx x_\infty(t)[1+\phi(t)+\mu(t)]$, where $\mu(t)$ is smaller than $\phi(t)$, one finds

$$\boxed{x(t) \approx \left(\frac{t}{\Lambda}\right)^{\frac{1}{n-1}} \left[1 + \frac{\Omega}{n-1}\left(\frac{\Lambda}{t}\right)^{\frac{n}{n-1}} - \frac{n\Omega^2}{2(n-1)^2}\left(\frac{\Lambda}{t}\right)^{\frac{2n}{n-1}}\right].} \tag{1.19}$$

Naturally, the perturbative parameter in this regime is

$$\phi(t) = \frac{\Omega}{n-1}\left(\frac{\Lambda}{t}\right)^{\frac{n}{n-1}}, \tag{1.20}$$

and thus

$$x(t) \approx x_\infty(t)\left[1+\phi(t)-\frac{n}{2}\phi^2(t)\right], \tag{1.21}$$

i.e., $\mu(t) = -n\phi^2(t)/2$.

EXERCISE 1-6. Derive analytical expressions for $x(t)$ to order $\delta^3(t)$ and $\phi^3(t)$.

EXERCISE 1-7. Consider the equation obtained by generalizing Eq. (1.13) to $x^n(t) = \Omega + t x^m(t)/\Lambda$ with $m < n$, and develop its approximate solutions for both large-t and small-t limits.

In FIG. 1.3 we show an example of the solution of Eq. (1.13) with $n = 6$ and $\Lambda = \Omega = 1$, for which $t_{\text{asp}} \approx 0.26$. The exact solution of Eq. (1.13) can be numerically constructed, for example via the iterative algorithm

$$\boxed{x^{(j+1)}(t) = \left(\Omega + \frac{t x^{(j)}(t)}{\Lambda}\right)^{1/n}, \qquad j = 0, 1, 2, \cdots,} \tag{1.22}$$

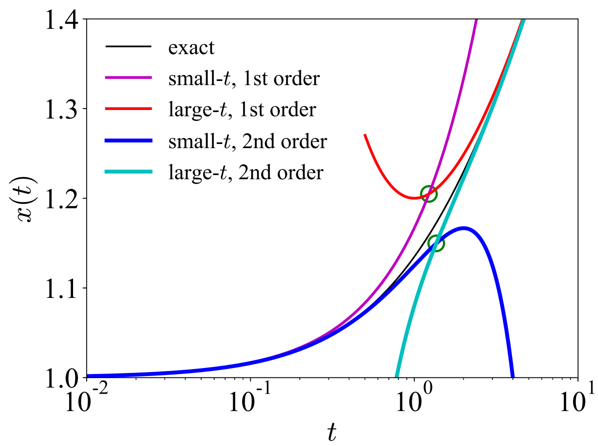

FIG. 1.3: Solution of Eq. (1.13), $n = 6, \Lambda = \Omega = 1$.

starting from an initial value $x^{(0)}(t)$. It is evident from the figure that after combining the asymptotic solution with the small-t approximation, one can eventually construct the full solution of Eq. (1.13) (for example, the region bounded by the second-order small-t approximation shown in blue and the second-order asymptotic solution shown in cyan). Moreover, the accuracy can always be improved by including higher-order terms in both the small-t and large-t expansions.

In addition, the convergence structure of the solution (either in the small-t or in the asymptotic regime) is commutative in the sense that the deviation from the exact one is commutative. For the special case $n = 3$, the second-order term in the small-t expansion vanishes, indicating that even higher-order terms may play a similar role in determining the solution, and these terms must therefore be examined carefully. Furthermore, the asymptotic solution is inversely correlated with t. In this sense, we have effectively developed a perturbative expansion in the small parameter $1/t$ when t itself is large. The perturbative relation in terms of t and/or $1/t$ is referred to as the duality of the problem. Finally, the intermediate region (for example, the region bounded by the magenta (blue) and red (cyan) curves) is generally non-perturbative. The solution in this region typically depends on a numerical prescription such as (1.22), although the size of this intermediate region decreases as higher-order perturbative corrections are included. For example, the applicable region obtained by matching the first-order theories through $x^{\text{1st}}_{\text{small-}t}(t) \approx x^{\text{1st}}_{\text{large-}t}(t)$ is approximately $0 \lesssim t \lesssim 1.23$, whereas for the second-order theories it extends to about $0 \lesssim t \lesssim 1.36$ (see the green circles in FIG. 1.3). Although the curves are much closer to the exact one, the improvement is limited [*1-2*].

EXERCISE 1-8. Compute the area formed by the exact curve, the large-t curve, and the small-t curve to first and second order as shown in FIG. 1.3, respectively.

§1.4 Divide-and-conquer and the Master Theorem

Assume that the time required to solve a problem of size n is denoted by $T(n)$, and suppose that $T(n)$ satisfies the recursive relation

$$\boxed{T(n) = aT(n/b) + \mathcal{O}(n^d),} \tag{1.23}$$

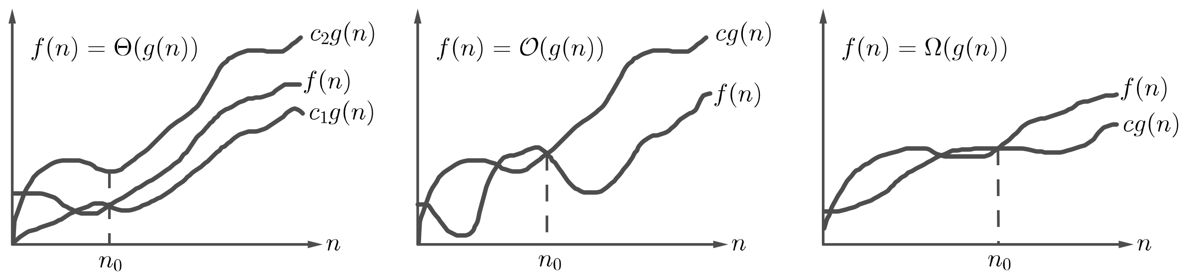

FIG. 1.4: Graphic illustrations of the Θ, $\mathcal{O}$, and Ω notations.

that is, a problem of size n is solved by recursively handling a subproblems, each of size n/b, and then combining their results in $\mathcal{O}(n^d)$ time, with $a, b, d > 0$ (see FIG. 1.4). Consequently,

$$T(n) = \begin{cases} \mathcal{O}(n^d), & d > \log_b a, \\ \mathcal{O}(n^d \log n), & d = \log_b a, \\ \mathcal{O}(n^{\log_b a}), & d < \log_b a, \end{cases} \tag{1.24}$$

where $\log \equiv \log_2$. The relation (1.24) is known as the master theorem [∗1-3∗], and a schematic illustration is provided in FIG. 1.5. In the above, the notations $\mathcal{O}$, and similarly Θ and Ω, are defined as

$$\Theta(g(n)) = \{f(n) : \exists c_1, c_2, n_0 > 0 \text{ such that } 0 \le c_1 g(n) \le f(n) \le c_2 g(n) \text{ for } n \ge n_0\}, \tag{1.25}$$
$$\mathcal{O}(g(n)) = \{f(n) : \exists c, n_0 > 0 \text{ such that } 0 \le f(n) \le c\,g(n) \text{ for } n \ge n_0\}, \tag{1.26}$$
$$\Omega(g(n)) = \{f(n) : \exists c, n_0 > 0 \text{ such that } 0 \le c\,g(n) \le f(n) \text{ for } n \ge n_0\}. \tag{1.27}$$

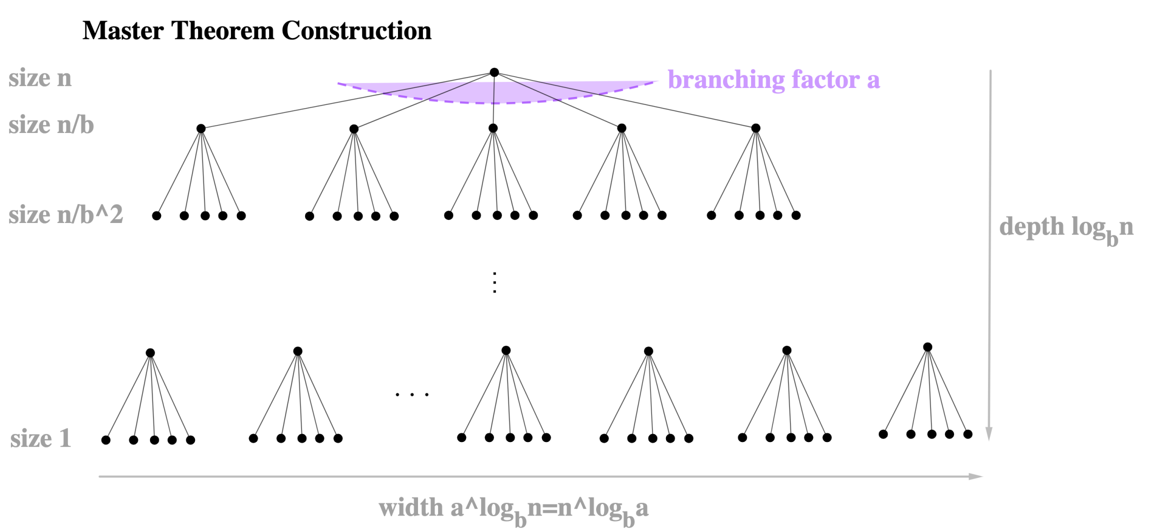

FIG. 1.5: Schematic representation of the divide-and-conquer method.

FIG. 1.6 displays the n-dependence of four functions, namely $n^2, 5n, 2^n$, and $\log n$. One observes that the growth rate of 2^n is significantly faster than that of the other three functions. Moreover,

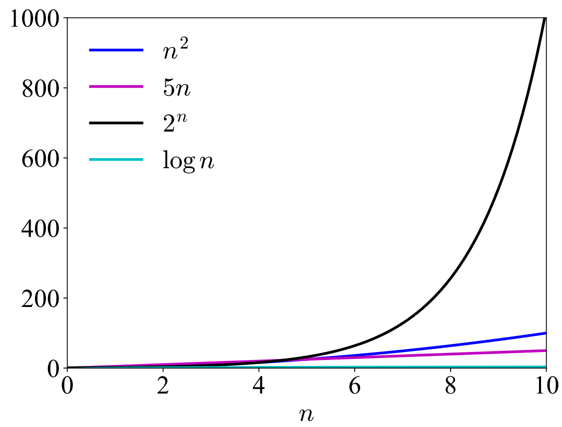

FIG. 1.6: Examples of functions $f(n)$ with different computational complexities.

although for small n the function $5n$ exceeds n^2, the quadratic function becomes dominant once $n \geq 5$. In contrast, the logarithmic function $\log n$ grows the slowest among them.

EXERCISE 1-9. Consider sorting m numbers stored in the array A by first finding the smallest element of A and exchanging it with the element stored in a_1. Then find the second smallest element of A, and exchange it with a_2. Continue in this manner for the first $m-1$ elements of A. This procedure is known as selection sort. Determine the best-case and worst-case running times.

EXERCISE 1-10. Solve the following recursive relations:

$T(n) = 2T(n/2) + n^4$, $T(n) = T(7n/10) + n$,
$T(n) = 16T(n/4) + n^2$, $T(n) = 7T(n/3) + n^2$,
$T(n) = 7T(n/2) + n^2$, $T(n) = 2T(n/4) + \sqrt{n}$,
$T(n) = T(n-2) + n^2$, $T(n) = 4T(n/3) + n\log n$,
$T(n) = 3T(n/3) + n/\log n$, $T(n) = 4T(n/2) + n^2\sqrt{n}$,
$T(n) = 3T(n/3-2) + n/2$, $T(n) = 2T(n/2) + n/\log n$,
$T(n) = T(n-1) + 1/n$, $T(n) = T(n-1) + \log n$,
$T(n) = T(n-2) + 1/\log n$, $T(n) = \sqrt{n}T(\sqrt{n}) + n$.

§1.5 Example 1: Fibonacci Series

Exploring computational complexity is one of the central tasks in algorithm analysis. For example, the Fibonacci sequence is defined by

$$F_n = \begin{cases} F_{n-1} + F_{n-2}, & n > 1, \\ n, & n = 1, 0. \end{cases} \tag{1.28}$$

The Fibonacci numbers grow almost as fast as powers of 2. For instance, F_{30} exceeds one million, and F_{100} already contains 21 digits. The closed-form expression for the Fibonacci numbers is

$$F_n = \frac{1}{\sqrt{5}}\left[\phi^n - (1-\phi)^n\right] = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right], \tag{1.29}$$

where ϕ is the constant defined by

$$\phi = \frac{1+\sqrt{5}}{2} = 1 + \cfrac{1}{1+\cfrac{1}{1+\cdots}}. \tag{1.30}$$

For large n, the following asymptotic approximation holds,

$$F_n \approx \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n. \tag{1.31}$$

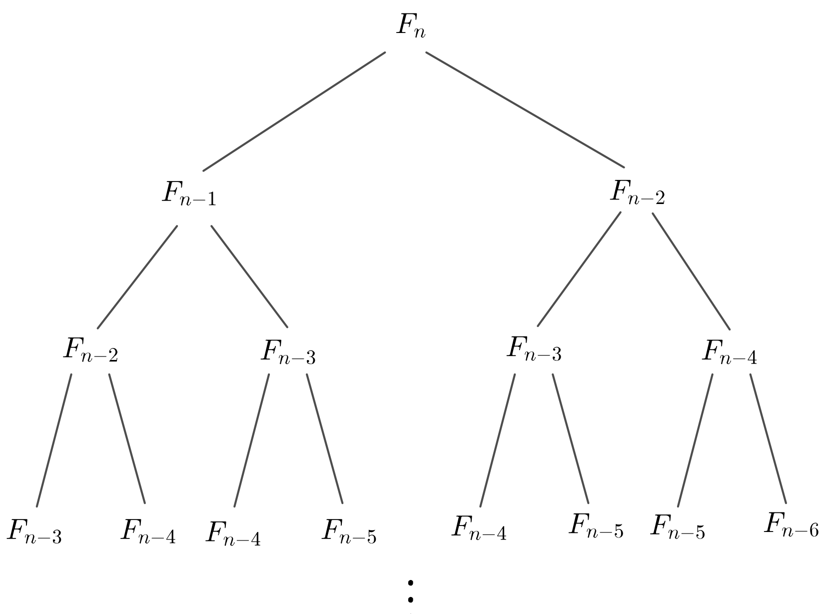

FIG. 1.7: Structure of the Fibonacci numbers.

According to the recursive definition, one immediately obtains the relation for the number of computational steps: $T(n) \le 2$ for $n \le 1$, and $T(n) = T(n-1) + T(n-2) + 3$ for $n > 1$, which implies that $T(n) \ge F_n$. Therefore, the running time of the naive recursive algorithm grows as fast as the Fibonacci numbers themselves, i.e., $T(n)$ is exponential in n. Consequently, the algorithm becomes impractically slow except for very small values of n. In summary, the naive recursive algorithm is correct but extremely inefficient. FIG. 1.7 illustrates the cascade structure of the recursion, and the poor performance arises because many intermediate values are recomputed repeatedly.

To design a more efficient algorithm, one should store intermediate results as soon as they are computed, namely the values $F_0, F_1, \cdots, F_{n-1}$. With this improvement, the number of computational steps becomes linear in n.

Using the asymptotic relation (1.31), the Fibonacci number for large n can be expressed as [∗1-4∗]

$$\boxed{F_n \approx 2^{-(1/2)\log 5 + 0.694n},} \tag{1.32}$$

which shows that it is approximately $0.694n$ bits long and can easily exceed 32 bits as n increases.

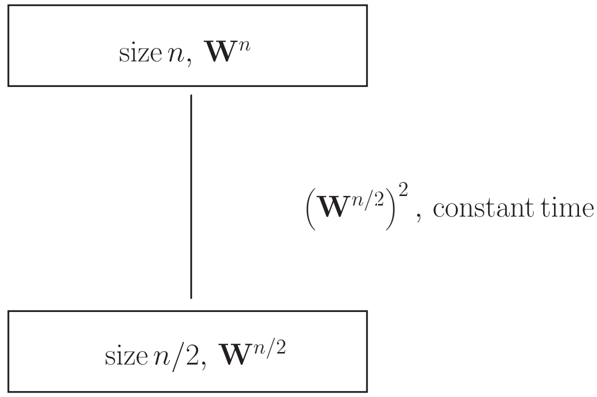

FIG. 1.8: Using matrix multiplication to compute the Fibonacci numbers.

Arithmetic operations on arbitrarily large integers cannot be carried out in constant time. Therefore, earlier running-time estimates must be revisited and made more precise.

In fact, adding two n-bit integers requires time roughly proportional to n. Thus, the naive algorithm performs about F_n additions, and the actual number of basic operations is roughly proportional to nF_n. Similarly, the improved dynamic programming algorithm requires on the order of n^2 basic steps, which remains polynomial in n. Finally, the matrix-based method involves roughly $n \log n$ steps. Whether the matrix-based algorithm is faster than the improved algorithm depends on whether n-bit integers can be multiplied in time faster than $\mathcal{O}(n^2)$. For example, the divide-and-conquer multiplication algorithm achieves complexity $\mathcal{O}(n^{1.59})$. Thus, we reduce the running time from exponential to polynomial: a dramatic improvement. This example clearly demonstrates that choosing the right algorithm makes all the difference.

Even more efficient algorithms exist for computing Fibonacci numbers. Observe that

$$\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} F_{n-1} \\ F_n \end{pmatrix} = \cdots = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}, \tag{1.33}$$

so computing F_n reduces to raising this 2×2 matrix to the nth power. Denote this matrix by $\mathbf{W}$. The computation of $\mathbf{W}^n$ can be carried out in $\mathcal{O}(\log n)$ time via repeated squaring, since $\mathbf{W}^n = (\mathbf{W}^{n/2})^2 = \cdots$ when n is a power of 2. Hence, the number of arithmetic operations required by the matrix-based algorithm is $\mathcal{O}(\log n)$.

From the viewpoint of recurrence relations, once $\mathbf{W}^{n/2}$ is known, $\mathbf{W}^n$ can be obtained in constant time (for matrix multiplication), leading to

$$\boxed{T(n) = T(n/2) + \mathcal{O}(1),} \tag{1.34}$$

with $a = 1$, $b = 2$, and $d = 0$. According to the master theorem, this implies $T(n) = \mathcal{O}(\log n)$. See FIG. 1.8 for a schematic illustration of the algorithm.

§1.6 Example 2: Fast Matrix Multiplication

We apply a divide-and-conquer strategy to matrix multiplication. Let $\mathbf{Z} = \mathbf{XY}$, where $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{n \times n}$, and for simplicity assume that n is a power of 2. The standard definition of matrix multiplication is [∗1-5∗]

$$Z_{ij} = \sum_{k=1}^{n} X_{ik} Y_{kj}, \tag{1.35}$$

which has a complexity $\sim \mathcal{O}(n^3)$. See FIG. 1.9 for a sketch of matrix multiplication.

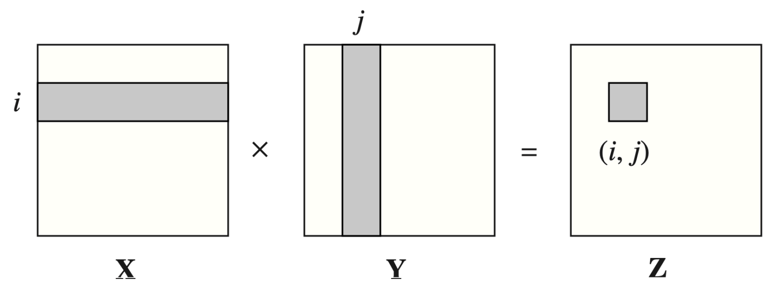

FIG. 1.9: Sketch of matrix multiplication.

To derive a divide-and-conquer formulation, decompose **X** and **Y** into block matrices:

$$\mathbf{X} = \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix}, \qquad \mathbf{Y} = \begin{pmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{G} & \mathbf{H} \end{pmatrix}, \tag{1.36}$$

where each block is of size $n/2 \times n/2$. Then

$$\mathbf{XY} = \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix} \begin{pmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{G} & \mathbf{H} \end{pmatrix} = \begin{pmatrix} \mathbf{AE}+\mathbf{BG} & \mathbf{AF}+\mathbf{BH} \\ \mathbf{CE}+\mathbf{DG} & \mathbf{CF}+\mathbf{DH} \end{pmatrix}. \tag{1.37}$$

Thus, computing the size-n product **XY** reduces to recursively computing eight products of size $n/2$: **AE**, **BG**, **AF**, **BH**, **CE**, **DG**, **CF**, **DH**, together with $\mathcal{O}(n^2)$ additional matrix additions. The running time therefore satisfies the recurrence

$$T(n) = 8T(n/2) + \mathcal{O}(n^2). \tag{1.38}$$

By the master theorem, since $a = 8$, $b = 2$, and $d = 2$, we obtain $T(n) = \mathcal{O}(n^3)$.

EXERCISE 1-11. For $\mathbf{A} \in \mathrm{R}^{m\times d}$ and $\mathbf{B} \in \mathrm{R}^{d\times n}$, determine the number of operations for **AB**.

However, a clever rearrangement of terms yields

$$\mathbf{XY} = \begin{pmatrix} \mathbf{P}_5 + \mathbf{P}_4 - \mathbf{P}_2 + \mathbf{P}_6 & \mathbf{P}_1 + \mathbf{P}_2 \\ \mathbf{P}_3 + \mathbf{P}_4 & \mathbf{P}_1 + \mathbf{P}_5 - \mathbf{P}_3 - \mathbf{P}_7 \end{pmatrix}, \tag{1.39}$$

where

$$\mathbf{P}_1 = \mathbf{A}(\mathbf{F}-\mathbf{H}),\ \ \mathbf{P}_2 = (\mathbf{A}+\mathbf{B})\mathbf{H},\ \ \mathbf{P}_3 = (\mathbf{C}+\mathbf{D})\mathbf{E},\ \ \mathbf{P}_4 = \mathbf{D}(\mathbf{G}-\mathbf{E}), \tag{1.40}$$

$$\mathbf{P}_5 = (\mathbf{A}+\mathbf{D})(\mathbf{E}+\mathbf{H}),\ \ \mathbf{P}_6 = (\mathbf{B}-\mathbf{D})(\mathbf{G}+\mathbf{H}),\ \ \mathbf{P}_7 = (\mathbf{A}-\mathbf{C})(\mathbf{E}+\mathbf{F}). \tag{1.41}$$

This formulation requires only seven recursive multiplications of size $n/2$. The corresponding recurrence relation becomes

$$T(n) = 7T(n/2) + \mathcal{O}(n^2). \tag{1.42}$$

Applying the master theorem gives

$$\boxed{\mathcal{O}(n^{\log 7}) \approx \mathcal{O}(n^{2.81}).} \tag{1.43}$$

This method is known as Strassen's algorithm. Currently, the best improvements in this direction, beginning with the Coppersmith–Winograd approach [∗1-6∗] and its optimal extensions, achieve a complexity of approximately $\mathcal{O}(n^{2.37})$.

§1.7 Estimating the Energy of Atomic Bomb of 1945

Let us apply dimensional analysis to estimate the energy of the first atomic bomb explosion. The solution depends on the energy $E \sim \mathrm{J} \sim \mathrm{kg}\cdot\mathrm{m}^2/\mathrm{s}^2$, the density of the gas $\rho \sim \mathrm{kg}/\mathrm{m}^3$, and the specific heat γ (dimensionless). Combining the radius $R \sim$ m and the time $t \sim$ s, one can construct the following dimensionless quantity [∗1-7∗]

$$\Pi = Et^2/\rho R^5. \tag{1.44}$$

t (ms)	0.10	0.24	0.38	0.52	0.80	0.94	1.08	1.22	1.50	1.65
R (m)	11.1	19.9	25.4	28.8	34.2	36.3	38.9	41.0	44.4	46.0
/	1.79	1.93	3.53	3.80	4.07	4.34	15.0	25.9	34.0	53.0
/	46.9	48.7	61.1	62.9	64.3	65.6	106.5	130.0	145.0	175.0

TAB. 1.1: Series of time and radius of the atomic bomb explosion.

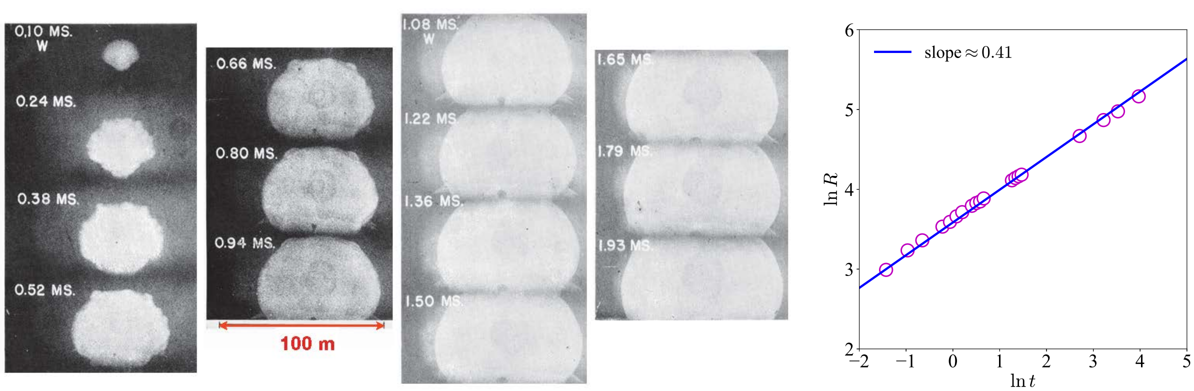

FIG. 1.10: Left: series of explosion of the atomic bomb. Right: scatters of $\ln R$ and $\ln t$ and its linear fitting.

Consequently, the relation between R and t can be written in the form

$$\boxed{R = \left(E/\rho\right)^{1/5} t^{2/5}\Phi(\gamma),} \tag{1.45}$$

where $\Phi(\gamma)$ is dimensionless. Therefore,

$$R \sim t^{2/5}, \tag{1.46}$$

or $\ln R \sim (2/5)\ln t$. See the left panel of FIG. 1.10 and TAB. 1.1 for images from the explosion sequence of the atomic bomb and the corresponding data for the time and radius, and the right panel of FIG. 1.10 for the fitting result for the relation between $\ln R$ and $\ln t$. It is found that the overall relation between $\ln R$ and $\ln t$ is linear with a slope of about 0.41, indicating that γ remains nearly constant during the explosion.

Solving for the energy E gives

$$\boxed{E = \Psi(\gamma)\frac{\rho R^5}{t^2},} \tag{1.47}$$

where $\Psi(\gamma) = 1/\Phi^5(\gamma)$ is another dimensionless quantity. Assume that $\Psi(\gamma)$ is around 1, and take $\rho \approx 1.25\,\mathrm{kg/m^3}$, with the average value $R^5/t^2 \approx 6.67 \times 10^{13}\,\mathrm{m^5/s^2}$. Consequently, $E \sim \rho R^5/t^2 \approx 8.34 \times 10^{13}\,\mathrm{J} \sim 10^{14}\,\mathrm{J}$, which is on the order of 1.7×10^7 kg TNT.

§1.8 Problems for This Lecture

Theoretical

1. One of the root of the equation $ax^2 + bx + c = 0$ with $abc \neq 0$ and $b > 0$ is given by

$$x^* = \frac{-b + \sqrt{b^2 - 4ac}}{2a} = \frac{b}{2a}\left(\sqrt{1 - \frac{4ac}{b^2}} - 1\right). \tag{1.48}$$

Assume that a is small in the sense $k = 4ac/b^2 \ll 1$, try to obtain the approximation for x^* from (1.48) by expanding the square root to order k^3 using the formula $\sqrt{1-k} \approx 1 - k/2 - k^2/8 - k^3/16$ for small k. The same result could also be obtained via firstly solving the equation $bx + c = 0$ and then adding some perturbation $p \sim k$ to the solution $-c/b$ as $x = -(c/b)(1 + p + \beta p^2)$. Determine the expression for p and the value of β. Another scheme to compute x to second order in k is to let p incorporate the relevant k-dependence at this order, i.e., set $p = mk + nk^2$. Show that by expanding x as a function of p and keeping terms up to k^2 then reproduces the same final expression for x as obtained using the previous expansion method.

2. Consider the equation $bx + c + ax^2 + (ax^2)^2 = 0$, where $|a| \approx 0$ is a small control parameter. Our goal is to construct approximate solutions using perturbative expansions. Two different schemes can be adopted: (1) first solve $bx + c = 0$ and then treat $ax^2 + (ax^2)^2$ as a perturbation; (2) first solve $bx + c + ax^2 = 0$ and then treat $(ax^2)^2$ as the perturbation. Show that, in both approaches, the solution up to second order can be expressed in the form $x \approx x_0^{(s)}(1 + k_{(s)} + \phi_{(s)} k_{(s)}^2)$ with $s = 1, 2$, where the corresponding coefficients are

$$\text{scheme 1: } x_0^{(1)} = -c/b, \quad k_{(1)} = -a x_0^{(1)}/b, \quad \phi_{(1)} = 2 + c; \tag{1.49}$$

$$\text{scheme 2: } x_0^{(2)} = \frac{-b + \sqrt{b^2 - 4ac}}{2a}, \quad k_{(2)} = -\frac{a^2 x_0^{(2),3}}{b + 2a x_0^{(2)}}, \quad \phi_{(2)} = 4 - \frac{a x_0^{(2)}}{b + 2a x_0^{(2)}}. \tag{1.50}$$

Show that scheme 2 yields a more accurate approximation than scheme 1. Argue that even if $|a|$ is not small the scheme 2 may still be good. What is term $\phi' k^3$? An illustrative example is presented in FIG. 1.11. Using the same strategy, further extend the perturbative construction to the generalized equation $bx + c + ax^2 + (ax^2)^\sigma = 0$ with $\sigma \geq 3$. This problem highlights the key idea of improving perturbative calculations through a more suitable choice of the zeroth-order approximation. A similar idea appears frequently in quantum-mechanical perturbation theory. If the

Hamiltonian is written as $H = H_0 + \lambda V$ with $H_0 = \mathbf{p}^2/2m$, then the usual perturbative expansion starts from the eigenvalue problem $H_0|n^{(0)}\rangle = E_n^{(0)}|n^{(0)}\rangle$, and treats λV as a small correction. However, one may absorb part of the interaction into an improved zeroth-order Hamiltonian, writing $H = \overline{H}_0 + \lambda\overline{V}$ with $\overline{H}_0 = H_0 + \lambda V_0$ and $\overline{V} = V - V_0$, so that the remaining perturbation is reduced. As a result, the corrections, $E_n \approx \overline{E}_n^{(0)} + \lambda\langle\overline{n}^{(0)}|\overline{V}|\overline{n}^{(0)}\rangle + \cdots$, are typically more accurate and exhibit better convergence. A standard example is the anharmonic oscillator with interactions such as $\mathcal{A}\mathbf{x}^3 + \mathcal{B}\mathbf{x}^4 + \cdots$, where one may take $\overline{H}_0 = \mathbf{p}^2/2m + 2^{-1}m\omega^2\mathbf{x}^2$, which, besides the kinetic term, includes a harmonic potential capturing the leading confining behavior. The remaining nonlinear terms are treated as $\lambda\overline{V}$, thereby improving the convergence of the perturbative expansion.

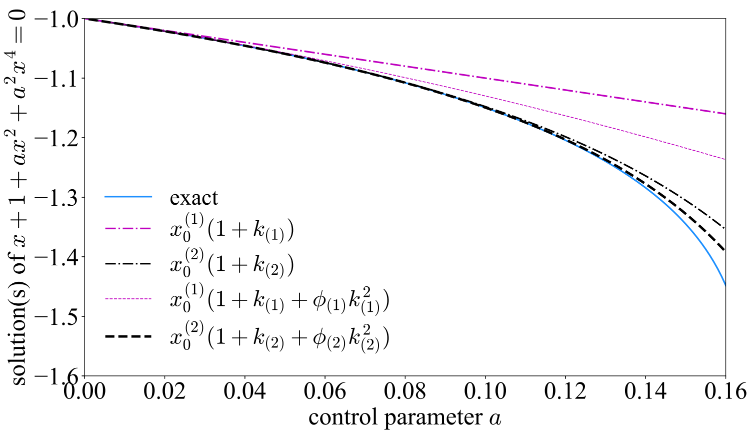

FIG. 1.11: Solution(s) of the equation $x + 1 + ax^2 + a^2x^4 = 0$.

3. *Show that the general perturbative expression for the period of a simple pendulum is given by

$$T(\chi_{\max}) = 2\pi\sqrt{\frac{\ell}{g}} \times \left[1 + \sum_{i=1}^{\infty}\frac{[(2i-1)!!]^2}{i!(2i)!!2^i}\left(\sum_{j=1}^{\infty}(-1)^{j-1}\frac{\chi_{\max}^{2j-1}}{2^{2j-1}(2j-1)!}\right)^{2i}\right]. \tag{1.51}$$

Use this formula to extract the coefficients of $\chi_{\max}^8$ and $\chi_{\max}^{10}$.
4. Prove the Pythagoras theorem via dimension analysis.
5. Use mathematical induction to prove the relation $F_n^2 - F_{n+1}F_{n-1} = (-1)^{n+1}$. Moreover, prove the following relations,

$$F_{k+n}F_{k-n} = F_k^2 - (-1)^{k-n}F_n^2,\quad F_{k+n} = F_kF_{n+1} + F_{k-1}F_n,\quad k,n\in\mathrm{Z}, \tag{1.52}$$

$$\sum_{k=0}^{n}\binom{n}{k}\left[f(k) + (-1)^{n-k}\sum_{s=0}^{k}\binom{k}{s}f(s)\right]F_{n-k} = 0,\quad n\in\mathrm{N}^+, \tag{1.53}$$

where f is some fixed function.
6. This problem concerns the computational complexity [∗1-8∗].
   (a) Prove the relations $f(n) = \Theta(f(n)), f(n) = \mathcal{O}(f(n)), f(n) = \Omega(f(n))$.
   (b) Let $f(n)$ and $g(n)$ be asymptotically nonnegative functions. Using the basic definition of Θ-notation, prove that $\max[f(n), g(n)] = \Theta(f(n) + g(n))$.
   (c) Is $2^{n+1} = \mathcal{O}(2^n)$? Is $2^{2n} = \mathcal{O}(2^n)$?

(d) In each of the following situations, indicate whether $f(n) = \mathcal{O}(g(n))$, or $f(n) = \Omega(g(n))$, or both ($f(n)$-left, $g(n)$-right, here and in the following $\log \equiv \log_2$ and $\ln \equiv \log_e$):

(i) $n-100$, $n-200$.

(ii) $n^{1/2}$, $n^{2/3}$.

(iii) $100n + \log n$, $n + (\log n)^2$.

(iv) $n \log n$, $10n \log 10n$.

(v) $\log 2n$, $\log 3n$.

(vi) $10 \log n$, $\log(n^2)$.

(vii) $n^{1.001}$, $n \log^2 n$.

(viii) $n^2/\log n$, $n \log^2 n$.

(ix) $n^{0.1}$, $\log^{10} n$.

(x) $(\log n)^{\log n}$, $n/\log n$.

(xi) $\sqrt{n}$, $\log^3 n$.

(xii) $\sqrt{n}$, $5^{\log n}$.

(xiii) $n2^n$, 3^n.

(xiv) $n!$, 2^n.

(e) Show that $k \log k = \Theta(n)$ implies $k = \Theta(n/\ln n)$.

(f) Let $f(n)$ and $g(n)$ be asymptotically positive functions. Prove or disprove each of the following conjectures:

(i) $f(n) = \mathcal{O}(g(n))$ implies $g(n) = \mathcal{O}(f(n))$.

(ii) $f(n) + g(n) = \Theta(\min(f(n), g(n)))$.

(iii) $f(n) = \mathcal{O}(g(n))$ implies $\log(f(n)) = \mathcal{O}(\log(g(n)))$, where $\log(g(n)) \geq 1$ and $f(n) \geq 1$ for all sufficiently large n.

(iv) $f(n) = \mathcal{O}(g(n))$ implies $2^{f(n)} = \mathcal{O}(2^{g(n)})$.

(v) $f(n) = \mathcal{O}(f^2(n))$.

(vi) $f(n) = \mathcal{O}(g(n))$ implies $g(n) = \Omega(f(n))$.

(vii) $f(n) = \Theta(f(n/2))$.

(g) Show that, if c is a positive real number, then $g(n) = 1 + c + c^2 + \cdots + c^n$ is:

(i) $\Theta(1)$ if $c < 1$.

(ii) $\Theta(n)$ if $c = 1$.

(iii) $\Theta(c^n)$ if $c > 1$.

(h) Show that the nth Fibonacci number satisfies the equality,

$$F_n = \frac{\phi^n - \widehat{\phi}^n}{\sqrt{5}} = \left\lfloor \frac{1}{2} + \frac{\phi^n}{\sqrt{5}} \right\rfloor, \ \phi = \frac{1+\sqrt{5}}{2}, \ \widehat{\phi} = \frac{1-\sqrt{5}}{2}, \tag{1.54}$$

which is to say that the nth Fibonacci number is equal to $\phi^n/\sqrt{5}$ rounded to the nearest integer. Thus, Fibonacci numbers grow exponentially. Here $\lfloor x \rfloor$ is the greatest integer less than or equal to x, and similarly $\lceil x \rceil$ is the least integer greater than or equal to x.

(i) Prove that the solution of $T(n) = T(n-1) + n$ is $\mathcal{O}(n^2)$.

(j) Prove that the solution of $T(n) = T(\lceil n/2 \rceil) + 1$ is $\mathcal{O}(\log n)$.

(k) Prove that the solution of $T(n) = 2T(\lfloor n/2 \rfloor + 17) + n$ is $\mathcal{O}(n \log n)$.

(l) Solve the recurrence $T(n) = 3T(\sqrt{n}) + \log n$ by making a change of variables.

(m) Argue that the solution to the recurrence $T(n) = T(n/3) + T(2n/3) + cn$, where c is a constant, is $\Omega(n \log n)$.

(n) Let $a \geq 1$ and $b > 1$ be constants, and let $f(n)$ be a nonnegative function defined on exact powers of b. Define $T(n)$ on exact powers of b by the recurrence

$$T(n) = \begin{cases} \Theta(1), & n = 1, \\ aT(n/b) + f(n), & n = b^i, \end{cases} \tag{1.55}$$

where i is a positive integer, prove

$$T(n) = \Theta\left(n^{\log_b a}\right) + \sum_{j=0}^{\log_b n - 1} a^j f\left(\frac{n}{b^j}\right). \tag{1.56}$$

(o) Let $a \geq 1$ and $b > 1$ be constants, and let $f(n)$ be a nonnegative function defined on exact powers of b. Prove that a function $g(n)$ defined over exact powers of b by

$$g(n) = \sum_{j=0}^{\log_b n - 1} a^j f\left(\frac{n}{b^j}\right) \tag{1.57}$$

has the following asymptotic bounds for exact powers of b:

(i) If $f(n) = \mathcal{O}(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $g(n) = \mathcal{O}(n^{\log_b a})$.

(ii) If $f(n) = \Theta(n^{\log_b a})$, then $g(n) = \Theta(n^{\log_b a} \log n)$.

(iii) If $af(n/b) \leq cf(n)$ for some constant $c < 1$ and for all sufficiently large n, then $g(n) = \Theta(f(n))$.

(p) Let $a \geq 1$ and $b > 1$ be constants, and let $f(n)$ be a nonnegative function defined on exact powers of b. Define $T(n)$ on exact powers of b by the recurrence

$$T(n) = \begin{cases} \Theta(1), & n = 1, \\ aT(n/b) + f(n), & n = b^i, \end{cases} \tag{1.58}$$

where i is a positive integer. Prove that the $T(n)$ has the following asymptotic bounds for exact powers of b:

(i) If $f(n) = \mathcal{O}(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \mathcal{O}(n^{\log_b a})$.

(ii) If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log n)$.

(iii) If $af(n/b) \leq cf(n)$ for some constant $c < 1$ and for all sufficiently large n, then $T(n) = \Theta(f(n))$.

Computational/Programming

7. Solve the equation $x^8(t) = 1 + t\,x(t)/2$ using approximated/numerical algorithms (like the iteration (1.22)).

8. There exist many series for estimating π, e.g.,

$$\sqrt{12}\sum_{k=0}^{\infty}\frac{(-3)^{-k}}{2k+1},\quad 2\sum_{k=0}^{\infty}\frac{k!}{(2k+1)!!},\quad -3+\sum_{k=1}^{\infty}\frac{k2^k(k!)^2}{(2k)!},\quad \left[\frac{2\sqrt{2}}{9801}\sum_{k=0}^{\infty}\frac{(4k)!(1103+26390k)}{(k!)^4 396^{4k}}\right]^{-1}, \tag{1.59}$$

as well as

$$\pi \approx 768\sqrt{2-\sqrt{2+\sqrt{2+\sqrt{2+\sqrt{2+\sqrt{2+\sqrt{2+\sqrt{2+\sqrt{2+1}}}}}}}}},\quad \pi = 3+\cfrac{1^2}{6+\cfrac{3^2}{6+\ddots}}. \tag{1.60}$$

Write codes to simulate π based on these formulas.

9. Assume the equation for χ is given by

$$\ddot{\chi}+\phi\chi-\frac{1}{6}\phi\chi^3=0,\ \ \phi>0. \tag{1.61}$$

If χ is a small quantity $\chi \ll 1$, the last term on the left hand side could be treated as a perturbation and in this situation analytically solve the above equation for $\chi(t)$. Then solve it when the last term is included numerically. What's the difference?

10. *Adopt the divide-and-conquer idea to design a fast algorithm for multiplication and then implement the matrix-multiplication-based algorithm for the Fibonacci series.

References and Notes for Lecture 1

[∗1-1∗] Some useful relations are $x=\chi\ell$, $U(x)=-mg\ell\cos\chi$, $E=-mg\ell\cos\chi_{\max}$, and

$$\cos\chi=1-2\sin^2\frac{\chi}{2},\ \ \cos\chi_{\max}=1-2\sin^2\frac{\chi_{\max}}{2},\ \ \sin\frac{\chi}{2}\Big/\sin\frac{\chi_{\max}}{2}=\sin\vartheta,\ \ \frac{1}{2}\cos\frac{\chi}{2}\mathrm{d}\chi=\sin\frac{\chi_{\max}}{2}\cos\vartheta\mathrm{d}\vartheta.$$

[∗1-2∗] The technique of expanding in a small parameter in one regime and in its inverse in another regime is widely used in physics. For example, in statistical physics one often encounters calculations at very low temperatures, where a low-temperature expansion in T/E (with T the temperature and E a fixed scale) is adopted. Conversely, at very high temperatures one may perform an expansion in E/T, which is known as the high-temperature expansion. Both techniques have historically played important roles in the understanding and development of phase transition theory.

[∗1-3∗] For the proof of the master theorem, see T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, 4th edition, MIT Press, 2022, section 4.6.

[∗1-4∗] See, e.g., S. Dasgupta, C. Papadimitriou, and U. Vazirani, *Algorithms*, McGraw-Hill Education, 2006, Chap. 1.

[∗1-5∗] For example, for two 2×2 matrices $\mathbf{A}$ and $\mathbf{B}$, their product is

$$\mathbf{AB}=\begin{pmatrix} a_{11}b_{11}+a_{12}b_{21} & a_{11}b_{12}+a_{12}b_{22}\\ a_{21}b_{11}+a_{22}b_{21} & a_{21}b_{12}+a_{22}b_{22}\end{pmatrix},$$

which involves $2^2\times(2+1)=12$ operations. For general n, the number of operations is $n^2\times(n+n-1)=2n^3-n^2\sim\mathcal{O}(n^3)$. Hence, the classical matrix multiplication algorithm has complexity $\mathcal{O}(n^3)$.

[∗1-6∗] D. Coppersmith and S. Winograd, *Matrix Multiplication via Arithmetic Progressions*, Journal of Symbolic Computation, **9**, 251 (1990).

[∗1-7∗] For the original discussions on this problem, see, G. Taylor, *The Formation of A Blast Wave by A Very Intense Explosion, I. Theoretical Discussion*, Proc. Roy. Soc. **A** 201, 159 (1950), for the general theory; *ibid*, *The Formation of A Blast Wave by A Very Intense Explosion, II. The Atomic Explosion of 1945*, Proc. Roy. Soc. **A** 201, 175 (1950), for the relevant application on the 1945 explosion of the atomic bomb.

[∗1-8∗] See mainly, e.g., T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, 4th edition, MIT Press, 2022.