

訳者まえがき

性能の良いシステムを開発するためには、相応の金銭的・時間的なコストを支払う必要があるが、世界を跨いだ協業の成果物と言える OSS (open source software) の力により、このコストは着実に下がってきてている。OSS が実際に研究課題・開発課題・ビジネス課題を解決するために十分な機能を提供していることは、OSS の応用実績を見れば自明だ。実際、OSS の応用は広く社会に行き渡っている。一方、その OSS を裏側で支えている“アルゴリズム”的理解についても同様に行き渡っているかというと、そうではない。

本書はそれらのアルゴリズムを理解するための平易な入門書である、Bradford Tuckfield 氏による *Dive Into Algorithms — A Pythonic Adventure for the Intrepid Beginner* の日本語訳である。この原著タイトルは 2021 年に変更されており、との題名は *Algorithms for the Adventurous — Creative Solutions to Computational (and Human) Problems* だった。どちらのタイトルも「アルゴリズム」が強調されているが、内容的には、各章の話題ごとに Python コードとその詳しい説明が付属しており、自分でコードを動かし、アルゴリズムがもたらす結果を目で見て確認できるようになっている。つまり、付属のコードを通じ、アルゴリズム自体だけではなく、そのアルゴリズムがどのように動くか、また、そのためにはどのようなコードを書けばよいのかを学ぶことができるのだ。

本書は、パフェ作りのレシピや所得税額の計算手順の例によりアルゴリズムを紹介するインストラクションから始まり、ボールをキャッチするアルゴリズムの設計、和算を含むアルゴリズムの歴史など、興味深い例を通じて、アルゴリズムとは何かを読者に問いかけていく。読者は、ある時は総理大臣に、またある時は郵政長官になり、直面する問題をアルゴリズム的に解決するにはどうすればよいかを学んでいく。勾配上昇法・降下法をはじめとするアルゴリズムに頻出する手法や、アルゴリズムの効率性を測る方法、ランダム性などアルゴリズムの基礎となる概念を、具体例を用いて理解できるように工夫されている。本書で基本的なアルゴリズムについて学ぶと、日常の至るところに潜んでいるアルゴリズムに親しみが湧き、さらには、アルゴリズム自身が持つ強力な力を理解できるようになる。

より高度なアルゴリズムについても、巡回セールスマン問題によって学ぶシミュレーテッドアニーリングや、決定木、ミニマックス法、テキストのベクトル化などを、事例とともに紹介している。現存するすべてのアルゴリズムを網羅的にカバーすることはもちろん不可能だが、専門家が毎日使用している人気のアルゴリズムをたくさん学ぶことができる。これらの学習を通して

じて、どのようなアルゴリズムが OSS、ひいてはわれわれの社会を支えているのかを理解できるようになるのだ。

書題をはじめとして、本書のさまざまな場面で「冒険」という言葉が出てくる。訳者の勝手な推測ではあるが、これはこの本が、奥深く、そして豊かなアルゴリズムの世界を旅する第一歩になるように、という著者の願いの表れだろう。実り多き冒険となるよう、アルゴリズムをどう設計するのか、その良し悪しをどう判断するか、そもそもアルゴリズムを利用すべきではない状況とはどんなものなのなど、実際にアルゴリズムを用いる際のアドバイスが随所に見受けられる。データサイエンスの会社を経営しながらオンライン小説サイトを運営する Tuckfield 氏らしく、原著は至るところに著者のユーモアが溢れているが、訳者の力不足により、原文の魅力をすべて伝えきれているかはわからない。それでも、この邦訳書を通じて、一人でも多くの読者がアルゴリズムの世界に興味を持ち、さらなる冒険に飛び出してくれたら、訳者としてこれ以上の喜びはない。

翻訳では、イントロダクションと 1~5 章を川上、6 章を高柳、7~11 章を武川が分担している。本書の翻訳・出版にあたり、全体を通じてサポートいただいた共立出版の山内千尋氏に、ここに記して謝意を表したい。この場をお借りして改めて心より感謝申し上げる。最後に、日夜本書の翻訳に没頭し、また暇を見つけては不動産投資に関わる他愛もない相談と美術品鑑賞にばかり時間を割いてしまう私に愛想を尽かすことなく、生活をともにし続けてくれる愛する妻に感謝を記し、訳者まえがきとしたい。

2022 年 6 月

訳者を代表して 武川文則

イントロダクション

アルゴリズムはどこにでも存在する。実際、あなたも今日すでにいくつかのアルゴリズムを意識せざとも実行しているはずだ。この本で、あなたはたくさんのアルゴリズムを知ることになる。それらは単純なものから複雑なものまで、また、有名なものからあまり知られていないものまでさまざまであるが、すべて興味深く、そしてすべて学ぶ価値のあるものだ。

この本で紹介する最初のアルゴリズムは、最も美味しいアルゴリズムもある。それはベリーグラノーラパフェを作るアルゴリズムで、図1に全体が説明されている。このタイプのアルゴリズムは、あなたにとっては「レシピ」と呼んだほうがしっくり来るかもしれない。しかし、これはドナルド・クヌース (Donald Knuth) による「アルゴリズム」の定義「特定の問題を解決するための一連の操作を与える、有限のルールの集合」に当てはまる。

ベリーグラノーラパフェ

作り方

1. 大きなグラスに 1/6 カップのブルーベリーを入れる
2. 1/2 カップのプレーンヨーグルトをブルーベリーの上にかける
3. 1/3 カップのグラノーラをヨーグルトの上にのせる
4. 1/2 カップのプレーンヨーグルトをグラノーラの上にかける
5. その上にイチゴをのせる
6. お好きなホイップクリームをトッピングしてでき上がり

図1 アルゴリズム：特定の問題を解決するための一連の操作を与える、有限のルールの集合

パフェ作り以外にも、アルゴリズムに支配される身近な例がある。毎年、アメリカ政府はすべての成人の市民にとあるアルゴリズムを実行させ、正しくできなかつた者を刑務所送りにしようと躍起になる。2017年には、何百万ものアメリカ人が図2に書かれているアルゴリズムを実行する務めを果たした。これは1040-EZと呼ばれている用紙から取ってきたものである。

どうして税金とパフェが同じだと言えるのだろうか？税金は義務的だし、数字が出てくるし、骨が折れる。そして世界中で嫌われている。パフェはたまに好みで作るもので、芸術的であり、簡単である。そして例外なく愛されている。唯一の共通点は、両方ともアルゴリズムに

所得税額の計算

	計算手順	記入欄
1	賃金、給料、およびチップの合計額。これらはフォーム W-2（源泉徴収票）の項目 1 に記載されている。W-2 も添付のこと。	
2	課税利子所得。合計が \$1,500 以上の場合、1040-EZ 申告書は利用できない。	
3	失業補償とアラスカ永久基金の配当（説明書を参照）。	
4	1, 2, 3 行を合算せよ。これが調整後総所得となる。	
5	もし、あなた（合算申告の場合はあなたの配偶者も含む）を扶養家族として申告する者がいるなら、下記の該当する項目にチェックを入れ、裏面のワークシートの総額を記入せよ。 <input type="checkbox"/> あなた <input type="checkbox"/> 配偶者 もし誰もあなた（合算申告の場合はあなたの配偶者も含む）を扶養家族として申告しないなら、独身者の場合は \$10,400 を、夫婦合算申告の場合は \$20,800 を記入せよ。説明は裏面を参照。	
6	第 4 行から第 5 行を減算せよ。もし第 5 行の額が第 4 行より多ければ 0 とする。この額があなたの課税所得である。	
7	フォーム W-2 とフォーム 1099 に記載された連邦所得税の源泉徴収額。	
8a	勤労所得額控除（EIC）（説明書を参照）。	
8b	非課税実戦手当に該当 <input type="checkbox"/>	—
9	第 7 行と第 8a 行を合計せよ。この額があなたの税額控除分を加えた納付済み税額である。	
10	税額。第 6 行の課税所得に対する税額を、説明書にある税額表から調べ、記入せよ。	
11	医療保険：国民の責任（説明書を参照） 1 年間維持 <input type="checkbox"/>	
12	第 10 行と第 11 行の金額を合計せよ。この額があなたの合計税額となる。	

図 2 所得税額の計算手順はアルゴリズムの定義に当てはまる¹⁾

よって支配される点である。

偉大なコンピューター科学者のドナルド・クヌースは、アルゴリズムを定義するだけではなく、それが「レシピ」「手続き」「回りくどい長々とした説明」とほぼ同じ意味であると述べた。1040-EZ フォームによる申告の場合、この手続きは 12 個のステップ（有限のリスト）からなり、これらのステップは 1 つ 1 つが操作（例えばステップ 4 の足し算や、ステップ 6 の引き算など）を指示する。これによって、ある特定のタイプの問題を解決するのだ。つまり、脱税によって投獄されるのを避けることである。パフェ作りの場合には、それぞれ操作を指示する 6 個のステップ（ステップ 1 でブルーベリーを入れる、ステップ 2 でヨーグルトをかけるなど）からなる。これらもまた、ある特定のタイプの問題を解決するための操作である。つまり、パフェを作る（もしくは食べる）ことである。

¹⁾ 訳注：この表はオリジナルを便宜上簡略化して訳したものである。

アルゴリズムをより深く知るにつれて、あなたもアルゴリズムがどこにでも存在することに気づき、それがいかに強力になりうるかを理解するようになるだろう。第1章では、ボールをキャッチするという人間が持つ驚くべき能力について説明し、その能力を可能にする人間の潜在意識に存在するアルゴリズムについて詳しく調べていく。その後、コードのデバッグ、バイキングでどのくらい食べるべきか、収益の最大化、リストのソート、タスクのスケジューリング、テキストの校正、チエスや数独で勝つためのアルゴリズムについて説明する。これらを通じて、いくつかの重要とされる属性に基づいてアルゴリズムの良し悪しを判断することを学んでいく。あなたは次第に、職人技とも呼べるような感覚や、さらにはアルゴリズムの芸術性を理解し始めるだろう。これによって、定量的な面が重視され、正確性が求められがちな作業に、創造性や個性を持たせることができるのだ。

誰のための本か？

この本は、アルゴリズムを学ぶための、Pythonコード付きのわかりやすい入門書である。以下のようないくつかの経験や知識があれば、この本の内容を最大限理解できるだろう。

プログラミング/コーディング この本の主要な例は、Pythonコードによって書かれている。

1つ1つのコードは、Python未経験者やあまりプログラミングの経験がない読者でも理解できるように、最大限丁寧に説明している。とはいえ、少なくともプログラミングの初步（例えば変数の割り当てや、`for`ループ、`if/then`文、関数の呼び出しなど）をある程度理解している必要がある²⁾。

高校数学 アルゴリズムは、方程式を解く、あるいは最適化や代数計算を行うといったように、数学と同様のゴールを持つことが多い。また、アルゴリズムは、論理や厳密な定義を必要とするという観点から、数学的な思考に関連した多数の原理原則を重んじる。本書での議論のうちいくつかは、代数、ピタゴラスの定理、円周率、そしてほんのわずかな基礎的な微積分など、数学領域に足を踏み入れる。しかし、内容が難解になりすぎないように配慮しているため、高校で学ぶ以上の数学は、本書では必要ない。

以上の前提知識を持っている読者であれば、本書の内容をすべて理解することができるだろう。本書は以下のような読者を想定して書かれている。

学生 本書の内容は、高校や学部生レベルにおけるアルゴリズム、コンピューターサイエンス、プログラミングの入門クラスに適している。

²⁾ 訳注：ここで述べられているトピックに加え、`while`ループやリストなども使用されている。これらのトピックや、Pythonの実行方法（スクリプトやコマンドプロンプト、またはnotebookなど）にあまり馴染みがない場合には、Pythonの入門テキストを適宜参照しながら読み進められたい。`numpy`やなどのサードパーティーモジュールも使用されているので、後の「環境の構築」節の訳注も参照のこと。

開発者・エンジニア すでにこれらの職に就いている人でも、Python を使えるようになりたい人や、コンピューターサイエンスの基礎を学びたい人、アルゴリズム的思考を使ってコーディングをより良くする方法について知りたい人は、この本から役に立つスキルを学べることだろう。

アマチュア愛好家 この本の真のターゲットはアマチュア愛好家である。アルゴリズムは人生のあらゆる場面に関わるので、誰もが自分の身の回りの世界をより良く知るためにヒントを何かしらこの本の中に見つけることができる。

本書の構成

この本はすべてのアルゴリズムのあらゆる側面をカバーしているわけではない。本書は入門書である。この本を読んだあと、あなたはアルゴリズムとは何かについて確かな感触を得るはずだ。例えば、重要なアルゴリズムをどう実装するかや、アルゴリズムの良し悪しを判断し、アルゴリズムのパフォーマンスを最適化するにはどうすればよいかがわかるようになる。さらには、プロフェッショナルが今最もよく使う数多のアルゴリズムに詳しくなるだろう。本書は以下の章からなる。

第1章：アルゴリズムで問題解決 この章では、人間がどのようにボールをキャッチするかについて考える。人間の行動を支配する潜在意識に埋め込まれたアルゴリズムが存在している証拠を見つけ、それを通じてアルゴリズムの有用性や、どのようにアルゴリズムを設計すればよいかを学んでいく。

第2章：歴史上のアルゴリズム 時空を旅しながら、古代エジプト人やロシアの農民がどのように掛け算をしていたか、古代ギリシア人がどのようにして最大公約数を見つけたかを学び、中世の日本の数学者から魔方陣を作る方法を教わる。

第3章：関数の丘と谷——最大化と最小化 勾配上昇法と勾配下降法を紹介する。これらは関数の最大値や最小値を見つけるシンプルな方法であり、最適化という多くのアルゴリズムにとって重要な目的のために使われる。

第4章：アルゴリズムを測る——ソートと探索 リストをソートし、リスト中にある要素を探索するための基礎的なアルゴリズムを紹介し、それらのアルゴリズムの効率性や計算速度を測る方法についても学ぶ。

第5章：数学に現れるアルゴリズム 連分数の作成や、平方根の計算、擬似乱数の生成といった数学に関連したアルゴリズムについて考える。

第6章：高度な最適化 最適解を見つけるための高度な手法であるシミュレーテッドアニーリングについて学ぶ。さらに、最適化問題の典型例である巡回セールスマン問題に取り組む。

第7章：幾何学　さまざまな幾何学的応用に役立つボロノイ図の作り方を学ぶ。

第8章：言語　単語間のスペースが欠落している英文を修正する方法と、あるフレーズの次に来る単語を予測して、フレーズを補完する方法について学ぶ。

第9章：機械学習　基礎的な機械学習手法である決定木について学ぶ。

第10章：人工知能　野心的なプロジェクトに取り組む。それは、進行中のボードゲームのプレイヤーに最強の指し手を教えるためのアルゴリズムを実装するプロジェクトである。攻略するボードゲームは「ドットアンドボックス」と呼ばれる有名なゲームである。

第11章：さらに冒険を続ける勇者へ　アルゴリズムに関する質問に対して本書の適切な章を案内するチャットボットを作るところから、うまくいけば100万ドルの懸賞金を獲得できる、数独にも関わる難問の話題まで、アルゴリズムの冒険を続ける勇者に向けたトピックで本書を締めくくる。

環境の構築

本書で紹介するアルゴリズムは Python を使って実装している。Python は無料のオープンソースのプログラミング言語であり、主要なプラットフォームで動かすことができる。以下では、Windows, macOS, Linux 上に Python をインストールする方法について説明する³⁾。

WindowsへのPythonのインストール

Windowsへのインストール方法を説明する。

1. Windows用の最新バージョンのPythonのページ (<https://www.python.org/downloads/windows/>)を開く(最後のスラッシュを忘れないように注意)。
2. ダウンロードしたいPythonリリースをクリックする。最新のリリースをダウンロードするには、[Latest Python 3 Release - 3.X.Y]と書かれているリンクをクリックすればよい。ここで、“3.X.Y”は最新のバージョン番号(例えば3.8.3など)である⁴⁾。本書のコードはPython 3.6とPython 3.8で動作確認を行っている。古いバージョンをダウンロードしたい場合は、[Stable Releases]セクションをスクロールし、ダウンロードしたいバージョンを選択する。

3) 訳注：ここでは公式サイトからのPythonのインストール方法が紹介されている。情報が古くなっている可能性があるので、必要に応じて最新の情報を確認するとよい。2020年後半以降に発売されたAppleシリコン搭載のMacでは、後述のサードパーティーモジュールのインストールとあわせて環境構築が簡単ではないケースがあつたようなので、最新の関連情報を確認されたい。モジュールのインストールも含め、環境構築が難しい場合には、Google Colaboratory (<https://colab.research.google.com/notebooks/welcome.ipynb?hl=ja>)のようなブラウザベースのサービスを利用して、本書のコードを試すこともできる。

4) 訳注：2022年3月現在、最新は3.10.2となっている。

3. ステップ2でリンクをクリックすると選択したリリースのページが開く。このページの[Files]セクションにある[Windows x86-64 executable installer]⁵⁾をクリックする。
4. ステップ3でインストーラのリンクをクリックすると，“.exe”ファイルがコンピューターにダウンロードされる。このインストーラファイルをダブルクリックして開くと、インストールが自動的に始まる。[Add Python 3.X to PATH]（Xは“8”などあなたがインストールしたリリースの数字）ボックスにチェックを入れ、[Install Now]をクリックし、デフォルトのオプションを選択する。
5. [Setup was successful]が表示されたら[Close]をクリックし、インストール完了である。

インストールが完了すると、コンピューター上に新しいアプリケーションが作られる。新しいアプリケーションの名前はPython 3.X（Xは今インストールしたバージョン）である。Windowsの検索ボックスに“Python”と入力してアプリケーションを検索し、クリックすると、Pythonのコンソールが開く。このコンソールにPythonコマンドを入力すると、コマンドが実行される。

macOSへのPythonのインストール

macOSへのインストール方法を説明する。

1. macOS用の最新バージョンのPythonのページ(<https://www.python.org/downloads/mac-osx/>)を開く(最後のスラッシュを忘れないように注意)。
2. ダウンロードしたいPythonリリースをクリックする。最新のリリースをダウンロードするには、[Latest Python 3 Release - 3.X.Y]と書かれているリンクをクリックすればよい。ここで、“3.X.Y”は最新のバージョン番号(例えば3.8.3など)である⁶⁾。本書のコードはPython 3.6とPython 3.8で動作確認を行っている。古いバージョンをダウンロードしたい場合は、[Stable Releases]セクションをスクロールし、ダウンロードしたいバージョンを選択する。
3. ステップ2でリンクをクリックすると、選択したリリースのページが開く。このページの[Files]セクションにある[macOS 64-bit installer]⁷⁾をクリックする。
4. ステップ3のインストーラのリンクをクリックすると，“.pkg”ファイルがコンピューターにダウンロードされる。このインストーラファイルをダブルクリックして開くと、

⁵⁾ 訳注：リリースによっては、64ビットマシン用のリンク名が「Windows installer (64-bit)」などへ変更されている。

⁶⁾ 訳注：2022年3月現在、最新は3.10.2となっている。

⁷⁾ 訳注：リリースによってはリンク名の変更やマシンに合わせたインストーラを選ぶ必要があり、3.10.2では“macOS 64-bit universal2 installer”が存在する。

インストールが自動的に始まる。デフォルトのオプションを選択する。

インストーラによって、コンピューター上に “Python 3.X” (X は今インストールしたバージョン) という名前のフォルダーが作成される。このフォルダー内の IDLE という名前のアイコンをダブルクリックすると、Python 3.X.Y シェルが開く。これが Python コンソールであり、ここで Python コマンドを自由に実行できる。

LinuxへのPythonのインストール

Linuxへのインストール方法を説明する。

1. 使用する Linux のバージョンが使用するパッケージマネージャーを決める。一般的なパッケージマネージャーの例として、yum や apt-get などがある。
2. Linux コンソール（ターミナルとも呼ばれる）を開き、次の 2 つのコマンドを実行する。

```
> sudo apt-get update  
> sudo apt-get install python3.8
```

もし yum やその他のパッケージマネージャーを利用しているなら、上記のコマンドの “apt-get” を “yum” など該当するものに置き換える。同様に、古いバージョンの Python をインストールしたい場合には、“3.8”（本書執筆時点での最新のバージョン番号）を、本書のテストに使用したバージョンの 1 つである 3.6 などの他のリリース番号に置き換える。逆に、最新のバージョンにしたい場合は、まず最新のバージョン番号を確認するために <https://www.python.org/downloads/source/> を開く。そのページに [Latest Python 3 Release - Python 3.X.Y] (3.X.Y がリリース番号) のようなリンクが表示されており、その最初の 2 つの数字 (3.X) が最新のバージョンである。これを上記のコマンドで利用する。

以下のコマンドを Linux コンソールで実行すると、Python が起動する。

```
python3
```

Python コンソールが Linux コンソールウィンドウで開き、Python コマンドを入力できる。

サードパーティーモジュールのインストール

本書のコードのいくつかは、Python 公式サイトからダウンロードしたコア Python に含まれない Python モジュールを利用している。これらのサードパーティーモジュールをコンピュート

ターにインストールするには、<http://automatetheboringstuff.com/2e/appendixa/> の説明を参照されたい⁸⁾.

まとめ

本書でアルゴリズムを学ぶことを通じて、世界中を旅し、歴史を何世紀も遡る。古代エジプト、バビロン、ペリクレスのアテネ、バグダッド、中世ヨーロッパ、日本の江戸、そして英領インド帝国で用いられた技術から、現代の息を呑むような技術に至るアルゴリズムの革新について探求していく。その際、われわれはいきなりは解けない問題や困難な制約を乗り越える新たな方法を見つけるよう迫られる。それらを通じて、古代科学の先駆者たちだけではなく、コンピューターを使ったりボールをキャッチしたりする現代の人々と、さらには、われわれの遺産を受け継ぎ発展させる未来のアルゴリズムユーザーやクリエーターと繋がっていくのだ。この本は、そうしたアルゴリズムとの冒険にあなたを導く。

⁸⁾ 訳注：リンク先のページは英語である。主に Python モジュールを管理するシステムである pip を用いてインストールを行う方法が説明されている。

目 次

第 1 章 アルゴリズムで問題解決	1
解析的アプローチ	2
ガリレイモデル	2
x を解く戦略.....	4
内なる物理学者	5
アルゴリズム的アプローチ	6
自分の首で考える	7
チャップマンアルゴリズムを適用する	10
アルゴリズムで問題を解決する	11
まとめ	13
第 2 章 歴史上のアルゴリズム	14
ロシア農民の掛け算	15
RPM を手で計算する	15
Python で RPM を実装する	19
ユークリッドの互除法	21
ユークリッドの互除法を手で実行する	22
Python でユークリッドの互除法を実装する	23
日本の魔方陣	24
Python で Luo Shu Square を作る	24
Python で久留島アルゴリズムを実装する	25
まとめ	38
第 3 章 関数の丘と谷——最大化と最小化	39
税率を設定する	40
正しい方向へのステップ	40
ステップをアルゴリズムへ	44

勾配上昇法への反論	45
極値の問題	47
教育と生涯所得	47
生涯所得の丘を勾配上昇法で登る	49
最大化から最小化へ	50
丘登りの一般例	53
アルゴリズムを使うべきではない場合	54
まとめ	55
第4章 アルゴリズムを測る——ソートと探索	56
挿入ソート	57
挿入ソートにおける挿入タスク	57
挿入タスクによってソートする	60
アルゴリズムの効率を測る	61
なぜ効率を追求するのか？	62
時間を正確に計測する	63
ステップ数を数える	64
よく知られた関数と比較する	67
理論的な精度をさらに上げる	69
ビッグオー記法を利用する	71
マージソート	72
マージ	73
マージからソートへ	75
スリープソート	79
ソートから探索へ	81
2分探索	82
2分探索の応用例	84
まとめ	86
第5章 数学に現れるアルゴリズム	87
連分数	87
ϕ の圧縮と伝達	88
さらに連分数について	90

連分数を作成する	91
小数から連分数へ	95
分数から累乗根へ	98
平方根	98
バビロニア人のアルゴリズム	98
平方根を求める	100
乱数生成アルゴリズム	101
ランダム性の可能性	101
線形合同法	102
PRNG を評価する	103
ランダム性のダイハードテスト	105
線形フィードバックシフトレジスター	107
まとめ	111

第 6 章 高度な最適化 112

巡回セールスマン問題	113
問題を設定する	114
知力 vs. 腕力	118
最近傍アルゴリズム	120
最近傍探索を実装する	120
さらなる改善を確認する	122
貪欲なアルゴリズム	125
温度関数を導入する	126
シミュレーテッドアニーリング	128
アルゴリズムをチューニングする	131
巡回ルートの顕著な悪化を回避する	134
巡回ルートのリセットを許可する	135
パフォーマンスをテストする	136
まとめ	139

第 7 章 幾何学 140

郵便局問題	140
三角形入門	143

大学院レベルの高度な三角形の特性	146
外心を探す	146
プロット機能を改善する	148
ドロネー三角形分割	150
逐次的ドロネー三角形分割	151
ドロネー三角形分割を実装する	154
ドロネー三角形分割からボロノイ図へ	159
まとめ	163
第8章 言語	164
なぜ言語アルゴリズムは難しいのか	164
スペース挿入アルゴリズム	165
単語リストを定義し、単語を検索する	166
複合語を扱う	168
有効かもしれない単語をチェックする	169
インポートされたコーパスを使用して有効な単語をチェックする	170
欠落したスペースを適切に復元する	172
フレーズ補完アルゴリズム	176
トークン化して n -gram を取得する	176
検索サジェストのための戦略	178
$n+1$ -gram の候補を探す	178
出現頻度に応じてサジェストを選択する	180
まとめ	182
第9章 機械学習	183
決定木	183
決定木を構築する	185
データセットをダウンロードする	185
データを見る	186
データを分割する	188
よりスマートに分割する	189
分割変数を選択する	192
深さを追加する	194

決定木を評価する	197
オーバーフィットの問題	198
決定木モデルの改良	201
ランダムフォレスト	202
まとめ	202
第 10 章 人工知能	204
ドットアンドボックス	205
ゲームボードを描く	206
ゲームを表現する	207
ゲームのスコアを計算する	208
ゲームツリーと勝ち方	210
ゲームツリーを構築する	211
ゲームに勝つ	215
拡張機能を追加する	219
まとめ	220
第 11 章 さらに冒険を続ける勇者へ	221
アルゴリズムをもっと使いこなす	222
チャットボットを作る	223
文書のベクトル化	225
ベクトルの類似性	227
より良く、より速くするために	230
野心的なアルゴリズム	230
最も深い謎を解く	233
謝 辞	235
索 引	237