データサイエンスで考える理数探究基礎

「未来に向かって」

Python 版

金谷　重彦 編著

付録 Google Colaboratory 使い方も収録！

国立大学法人
奈良先端科学技術大学院大学
NARA INSTITUTE of SCIENCE and TECHNOLOGY

ISBN 978-4-902874-05-1

データサイエンスで考える理数探究基礎

「未来に向かって」

Python 版

金谷　重彦 編著

付録 Google Colaboratory 使い方も収録！

ISBN 978-4-902874-05-1

まえがき

文部科学省・高等学校学習要領（平成 30 年告示）解説・理数編（平成 30 年 7 月）によると「２理数科の要点」の中で『「理数探究基礎」では，生徒の特性や実態に応じて観察，実験，調査等の手法や統計処理の方法などを含んだ探究を遂行する上で必要な知識及び技能を身に付けさせる。また，実際に探究を遂行することなどを通して，各教科等で学習した知識及び技能を再確認したり新たな意味を見いだしたり，他の生徒と共に探究の方針を考えたり議論したりして粘り強く探究に取り組む態度を身に付けさせる。』とある。また、「3　内容と範囲，程度」においては『観察，実験，調査等で得られたデータには誤差やばらつきがあることを考慮して，例えば，平均値，標準偏差，相関係数などの統計量を基に分析したり，推定や仮説検定，単回帰などの統計的手法を活用したりすることが考えられる。数学科や情報科で学習する統計的な内容と関連させながら，得られたデータから実験結果を予測して実験を行ったり，実験値と，理論値や数値シミュレーションから得られた結果とを比較したりして分析の質を高めることも重要である。』とのことである。

さらに読み進めると、「理数科における探究的な学習の指導のポイント」について、理系文系に関わらず、「総合的探究時間」について、『①課題の設定：体験活動などを通して、課題を設定し課題意識をもつ。②情報の収集：必要な情報を取り出したり収集したりする。③整理・分析：収集した情報を、整理したり分析したりして思考する。④まとめ・表現：気づきや発見、自分の考えなどをまとめ、判断し、表現する。』という探究の過程が示されている。

これは、まさに、データサイエンスと捉えることもできる。国が掲げる「超スマート社会」を実現するための基盤技術の柱として情報科学分野（IoT システム構築、ビッグデータ解析、AI など）があらゆる分野で急速に発展を遂げている。「超スマート社会」の実現についても、さまざまなデータ解析が必要とされており、理数探究基礎をその基礎科目として位置づけることもできる。

理数探究基礎で扱う、「①課題の設定、②情報の収集、③整理・分析、④まとめ・表現」について効率的にパソコンを用いて行えば、この科目もデータサイエンスと位置付けられる。奈良先端大では、奈良県立奈良北高等学校とともに「いきなり大学院」というプロジェクトとして、先取り学習というのを進めている。しかし、産業界の声が、ここで必要になる。啓林館「理数探究基礎：未来に向かって」を参考に実際の高校の課題について

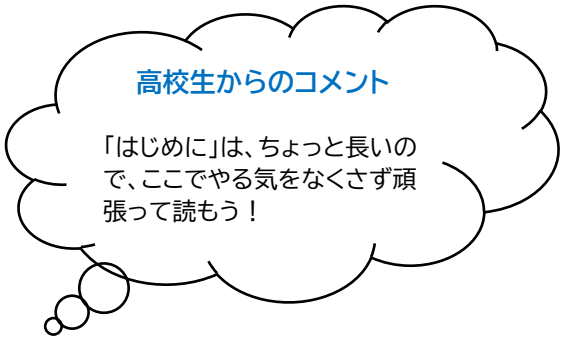
取り組んだところ、産業界が求めるデータサイエンスの側面で解決できる多くの課題があり、その紹介として本書を作成した。

目次

目次の「p.」は啓林館「理数探究基礎：未来に向かって」において参考にしたページです。

1. はじめに

探究の作法をきちんと学び、課題の設定方法や、数学・科学を活用した課題解決を学ぶのが目的です。自然界あるいは人間の社会活動についてみると、さまざまな研究テーマがあります。そこでは、課題を見つけ出す力が必要です。そして、これらを具体的に解きます。すなわち課題解決が必要になります。理数探究基礎では、このように、問題発見から解決法を学ぶ科目であると定義できます。この科目の特徴を整理すると、以下のようになります。

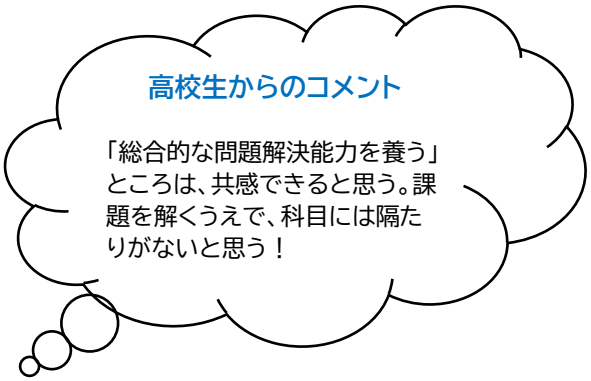


(1)「解答がない問題に挑む」

問題集を解く場合を想定すると、問題集には必ず解答があります。だから、解答へとたどり着く方法を理解すればいいのですが、「理数探究基礎」で扱う問題は、データから判断して、こう解釈できるという結論の導き方を勉強します。未知の問題を解いて、結論を導くにはどうしたらいいでしょうか。その一つに統計的判断なども活用し、仮説の妥当性を評価することも勉強するといいでしょう。奈良先端科学技術大学院大学などの機関では、いわゆる「課題」を発見し、「結論を導く」という研究を通して、新規発見・新規ものづくりを進めています。これを本書では体験していただきたく作成しました。

(2)「総合的な問題解決能力を養う」

高校生の皆さんは、国語、日本史、世界史、地理、化学、物理、生物、数学、英語、保健体育、美術、音楽などさまざまな科目を勉強します。その中で、理数探究では、これらの科目を融合して結論を導くことも大歓迎です。いろいろな科目で学んだ知識を活用する、あるいは、「理数探究基礎」をもとに、課題を設定し、これから学ぶ科目の関係を理解しながら、未知の問題に挑むというのもありでしょう。この科目の中で問題が解決できなくても、さまざまな教科を学ぶと、あるとき突然、解けることもあります。このような、問題意識を持ちながら、さまざまな教科を学ぶことも身に着けると、さらにそれぞれの教科を深く理解できると思います。なにか課題に突き当たった場合に、「これは、確か物理で学んだぞ！」とか、「健康の問題か、これは保健体育でいけるか！」、「食べ物の成分か、もしかしたら生物学、化学で対応できるか！」、「心地よさ



…音楽、国語？」などと色々な教科での問題として捉えるのも面白いかと思います。いろいろな経験を通して問題を解決すると人生に深みもでてきて、たかだか 100 年ほどの一個人の人生も面白いものになるでしょう。

(3)「データサイエンスへの誘い」

昨今、世の中では、AI、データサイエンス、データ DX などインフォマティクス(情報)が必要となっています。ではデータサイエンスとしての問題解決法はどのようになっているのでしょうか。

データサイエンスでの問題解決法の概略を**図 1** で説明します。

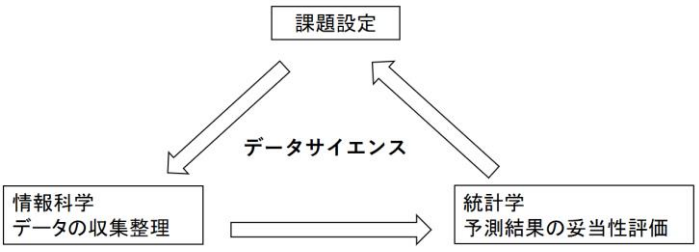


図 1 データサイエンスとは

まずはじめに、課題(仮説)を設定します。ここで必要とされることは、それぞれの分野の背景知識です。背景知識を活用し、いま、取り組もうとしている課題の新規性を検討します。その後、データを収集整理します。ここで、情報科学を活用し、より多くのデータを収集することで、データに潜むルール(仮説)の一般性を高めます。これらのデータを活用し、統計学を活用して、データに潜むルール(仮説)の妥当性を検証します。このようにして課題の解決を進めていきます。

実は、このような課題(仮説)解決法は、データサイエンスでのみ行われているのではなく、科学的問題解決の一般的な問題解決法です。非常に多くのデータを活用できる機会が増え、また、データ解析に必要なプログラミングも容易になったということから、データサイエンスとしての問題解決スキルが重要になってきたのだと思います。ですので、高校生の皆さんもプログラミングを楽しみながら、各自で設定した問題の解決を目指してみましょう。

高校生からのコメント

社会問題「電車内でのスマホ見すぎ問題を魅力ある広告により改善する」という課題を解くことを考えてみよう。

データ収集:「どの時間にどんな人がどれくらいの頻度でスマホを見ているかデータ収集をする。」

提示する対象者を検討する。朝はサラリーマンが、夕方は学生が多い。朝、夕方、夜で、それぞれに魅力的な広告を掲示するといった対策をして、同様にデータ収集をする。

広告の有無による、スマホの見すぎの改善をデータ収集により検討できるかもしれない。

2. 探究とは

［啓林館、理数探究基礎:未来に向かって、p.8］

「探究」とは、探し究めると書くように「何かを探しながら究めていくこと」です。いま、自ら課題を設定し、この解答(解釈)を探しながら究めるということになるでしょう。「探究」とは、皆さんのこれからの人生経験を深めていくうえでも、とっても重要な発想だと思います。毎日の生活の中に、「ああ、そうだったのか！」という解決があると、一生、探究なるものと付き合うことができると思います。これは、科学にのみあるものではなく、人文社会など社会の問題、小説などを読む場合にも「なるほど」と納得する発想だと思います。まずはじめに、科学者の探究の例を見てみましょう。

大隅良典博士は、2016 年、ノーベル生理学・医学賞を受賞しました。論文の形式をみると「探究」の方法が理解できると思うので、大隅良典博士の論文を読んでみましょう(**図 2.1**)。

● 科学研究論文を読んでみよう

本論文は、J. Biol. Chem., 256, 2079-2082 (1981)に掲載されました。論文のタイトルは、「Active Transport of Basic Amino Acids Driven by a Proton Motive Force in Vacuolar Membrane Vesicles of *Saccharomyces cerevisiae*」です。これを和訳してみよう。*Saccharomyces cerevisae* は「酵母菌の一種」、Vacuolar Membrane は「液胞膜」などと、コツコツ辞書を引きながらタイトルを訳すと「酵母 *Saccharomyces cerevisiae* の液胞膜小胞におけるプロトン駆動力により誘導される塩基アミノ酸の能動輸送」となり、酵母菌の液胞での、基本となるアミノ酸の輸送のメカニズムが述べられているのだとわかります。これを解明した論文で、ページ数にして 4 ページ、高校生の英語力があれば、辞書で単語の意味をコツコツと引きながら読み進めることができると思います。

高校生からのコメント

「高校 1 年の英語では、きついと思います。でも、説明を読めば、理解はできるかな？」

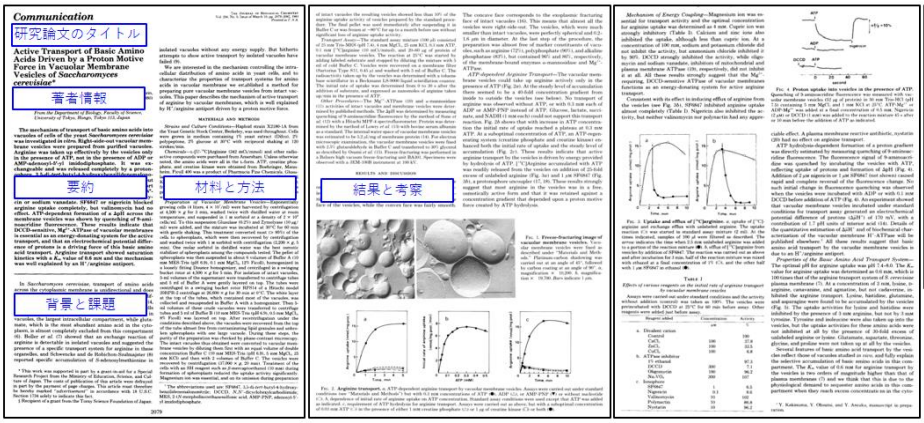


図 **2.1**　J. Biol. Chem., 256, 2079-2082 (1981)

タイトルの下に論文著者と所属が記載されている（著者情報）。続いて、要約、背景と課題設定、材料と方法、結果と考察、引用文献という項目で論文が作成されていることがわかると思います。

要約
本論文の要約が記述されています。

背景と課題設定
科学における探究では、今までの背景知識をもとに新たな発見が関わる課題が、背景と課題で説明されます。ここで、適に背景知識の論文を引用しながら説明がされているのが特徴です。これは、本論文を作成する上で背景知識をもとにした課題設定の妥当性を読者に明快にわかるようにするためです。

材料と方法
同様の実験を再現できることが目的で、実験に使った材料と実験方法が説明されています。以前に確立されている方法であれば引用文献を挙げながら説明をします。

結果と考察
結果では、実験で得られた結果を、適に図、表を用いて説明します。
考察では、結果をどのように解釈するかという視点でまとめます。他の論文との関係を説明するために引用文献を挙げながら説明をします。結論にさらに今後の展望について記述されることもあります。

引用文献
本論文で、引用した論文のリストです。

● 理数探究の進め方

発想・着想

社会現象・自然現象であれとにかく情報を分析しながら観察し、そこでなにが言えそうか考えます。例えば、「地球温暖化」というキーワードに興味をもてそうだと思ったら、まず公開されているデータがあるかどうか検討します。例えば、気象庁で公開されているデータなどがあれば、このデータを予備解析してみます。

高校生からのコメント

「理数探究の進め方」について、具体的例として「地球温暖化」を通して、発想から報告書作成までのプロセスについて理解できると思う。なるほど、こうやって理数探究をすればいいらしい！

予備解析

公開データをもとに、どのような生物の現象を見れば地球温暖化と生物の現象とを関係づけられそうか検討します。ここで、例えば、カエデなどの紅葉する植物の紅葉時期が、過去に比べて遅くなったことがデータ解析で得られたとします。

本実験

「温暖化となれば、紅葉する植物の紅葉時期がおくれる」などと仮説をたてて、ではどのような紅葉する植物の範囲でこの仮説が成り立つか、また、地球の温度上昇により、未来の紅葉時期など予測してもいいでしょう。このようにデータサイエンスをもとに統計解析を取り入れて、数理モデルの客観性を評価しましょう。

また、気温の変化と紅葉の関係が関与するか、例えば、紅葉していない葉を採取し、さまざまな温度で、一定の気温に保ち紅葉する状態を観察するのもありでしょう。ここで、紅葉させる条件を設定できれば、「人工紅葉法の開発研究」として工学的発想の研究へと展開できます。ここでも、設定された条件の妥当性を統計評価します。

報告書作成

実験で得た内容にもとに、おおまかに「**課題タイトル**、**研究者情報**、**要約**、**背景と課題設定**、**材料と方法**、**結果と考察**、**引用文献**」として報告書を作成します。課題タイトルは具体的内容がわかるように作成します。報告書を作成する上で、材料と方法、結果と考察では、具体的に、「結論として何が言えるか」を的確にわかりやすく説明できるか検討します。ここで、背景となる基礎知識、先行研究との関係も考慮し、出来る限り、実施した研究の客観性（だれが読んでも納得できる）を考慮して報告書を作成すればいいでしょう。また、解析結果で、例えば、外れ値（数理モデルなどから外れた値）については、これは実施した研究の限界を知るための重要な情報ですので、考察でなぜ特定のサンプルは外れ値になったのか、を考察すると、報告書がより客観的になります（**外れ値は考察の材料**）。報告書を作成するときに、適に外れ値があると、これを徹底的に考察して、論文を仕上げるのも一案です。外れ値を理由なく外しちゃだめよ。

社会現象あるいはマーケティングなどの分野でも同様の方法で研究が行われ、事業化へとつなげられています。つまり、文科系、理科系に関係なく、ほぼ同様に報告書を作成するわけですから、「探究」は理科系だけの課題であるというわけではありません。

● 研究論文の書き方

研究論文では、直接理解できる、あるいは、誤解を招かない文章（文学的表現はしない）で記述します。つまり、文学的言い回しは一切使わない。研究者が誤解なく理解できるように記述することが大切になります。だから、高校生の皆様でも、（辞書を引きながら）読むことができるし、直接伝わる英語で書かれています。ためしに、上記で説明した論文（J. Biol. Chem.,

256, 2079-2082 (1981))の要約(アブストラクト)を読んでみましょう。青色が研究目的、ピンク色が実験、緑色が結果、黒色が結論となっています。

The mechanism of transport of basic amino acids into vacuoles of cells of the yeast *Saccharomyces cerevisiae* was investigated in vitro. Right-side-out vacuolar membrane vesicles were prepared from purified vacuoles. Arginine was taken up effectively by the vesicles only in the presence of ATP, not in the presence of ADP or AMP-adenosyl-5'-yl imidodiphosphate. It was exchangeable and was released completely by a protonophore, 3,5-di-tert-butyl-4-hydroxybenzilidenemalononitrile (SF6847). The transport required M^{2+} ion but was inhibited by Cu^{2+}, Ca^{2+}, or Zn^{2+} ions. The transport activity was sensitive to the ATPase inhibitor N,N'-dicyclohexylcarbodiimide (DCCD), but not to oligomycin or sodium vanadate. SF6847 or nigericin blocked arginine uptake completely, but valinomycin had no effect. ATP-dependent formation of a ΔpH across the membrane vesicles was shown by quenching of 9-aminoacridine fluorescence. These results indicate that DCCD-sensitive, M^{2+}-ATPase of vacuolar membranes is essential as an energy-donating system for the active transport, and that an electrochemical potential difference of protons is a driving force of this basic amino acid transport. Arginine transport showed saturation kinetics with a Km, value of 0.6 mM and the mechanism was well explained by an H^{+}/arginine antiport.

科学論文の場合は以上のような構成になっています。探究をレポートとしてまとめる場合も大まかには上述のようにまとめるといいでしょう。

研究の流れ

1. 着想:観察する、情報を収集する
2. 予備解析:見通しをたてる、 データから仮説を検定する
3. 実験でさらに必要な情報を得る。
   実験できるかどうか考える:具体性を考える
   計算機実験:数値シミュレーション
4. 結論を導く
5. 報告書にまとめる、発表する

研究者はなぜ学術論文・学会などで発表するのでしょうか？学術論文誌への掲載プロセスをもとに考えてみましょう

学術論文誌に掲載されるプロセスを説明します。

(1) 研究者は作成した原稿を論文誌に投稿します(submission、学術誌の出版社に送ることです)。

(2) 編集者(Editor)は、その分野の他の研究者(査読者)に投稿された原稿の査読を依頼します。

(3) 査読者(Referee)は、原稿を詳細に査読(レビュー)し、原稿を評価します。このとき、原稿の独創性、原稿に追記すべき実験・解析などもチェックします。そこで、評価としては、受理(Accept)、条件的受理(Minor revision, Major revision)、不受理(Reject)の判断をコメントとともに編集者へ送ります。

(4) 査読者は複数人が対応するので、編集者はこれらの査読の内容をみて、どこを改善すべきか、などのコメントと最終判断としての受理、条件的受理、不受理の判定を研究者へ送ります。

(5) 研究者は、編集者からの査読結果をもとに、投稿した原稿に、追試験などを行い修正し、修正箇所を整理し、編集者へ送ります。

(6) ここで、編集者は、再度、査読者の意見を求め、ここで受理となれば、原稿は論文誌に掲載されます。

つまり、研究者の作成した原稿に、独創性、客観性があってはじめて論文として学術誌にて掲載されることになります。研究者の研究が学術誌に掲載されてはじめて、その研究者の研究の独創性と客観性が認められたことになります。学術誌に掲載されるあるいは学会発表をするまで、その研究は、一般には、まだ研究として認知されていないと考えるべきでしょう。つまり、研究者は、論文を作成することを考えながら実験・解析を進めることになります。論文を書かない研究者というのはありえません。

学術誌に査読プロセスがあるのはなぜでしょうか？

研究者から投稿された原稿に新規性があるかどうかが最も大切な点です。その上で、データを客観的に解析がなされているかが次に問われます。それで、原稿が受理され掲載された論文には、新規性と客観性が担保されたことになります。それで、査読プロセスが必要になるのです。

研究者からみると論文誌に掲載されている論文は、新規性があるので、自分の進めている研究との比較を行いながら、先行研究として掲載論文を勉強することになります。となると、研究者には、(a)新規に発見をする、あるいは新たに装置などを開発する発想力と(b)世界で発表されている研究内容を、学術論文誌を通して理解しておくことが求められます。というわけで、研究の発想に従って新規性のある結果を生み出す力と国際誌をコツコツ理解する力も求められます。「探究」という教科では、高校生にもいち早くこういった素養が鍛えることが求められているのでしょう。

学術論文ではなぜ引用文献が必要なのでしょうか？

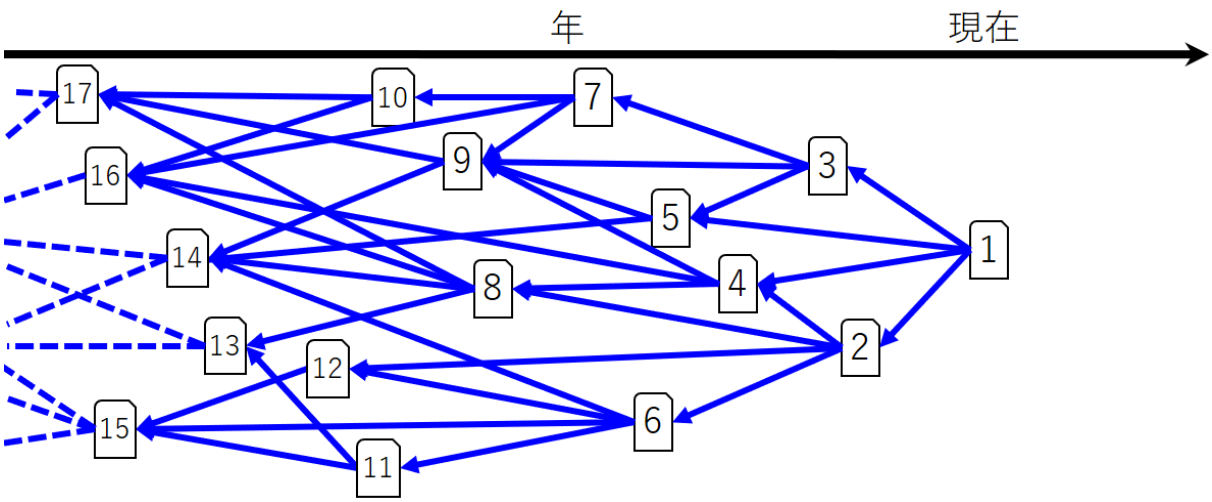


図 2.2：引用文献を通した学術論文の関係を表した概念図

いま、最新の学術論文を 1 とします（**図 2.2**）。1 の論文では、2,3,4,5 の論文を引用しています。また学術論文 2 は 4,6,8,12 を引用しています。このように、学術論文が引用した学術論文を矢印で表しました。**図 2.2** をみてわかることは、学術論文 1 の新規発見あるいは新規開発事例が、過去どのような研究から成り立っているかがわかります。査読の過程で適切な学術論文の引用がなされているかコメントされることもあります。ある研究者が進めている研究が 1 と近かったとします。そのとき、それぞれの学術論文から引用される学術論文（被引用論文）をコツコツと理解すれば、自分の研究の立ち位置がわかります。

また、このような引用システムがあれば、世界中の研究者の研究を比較検討ができます。その上で、「今進めている研究の新規性はなにか？」をつねに考えながら研究を進めることができます。近年、インターネットの普及により、学術論文を検索するサイトも提供されています。その一つに pubmed があります。

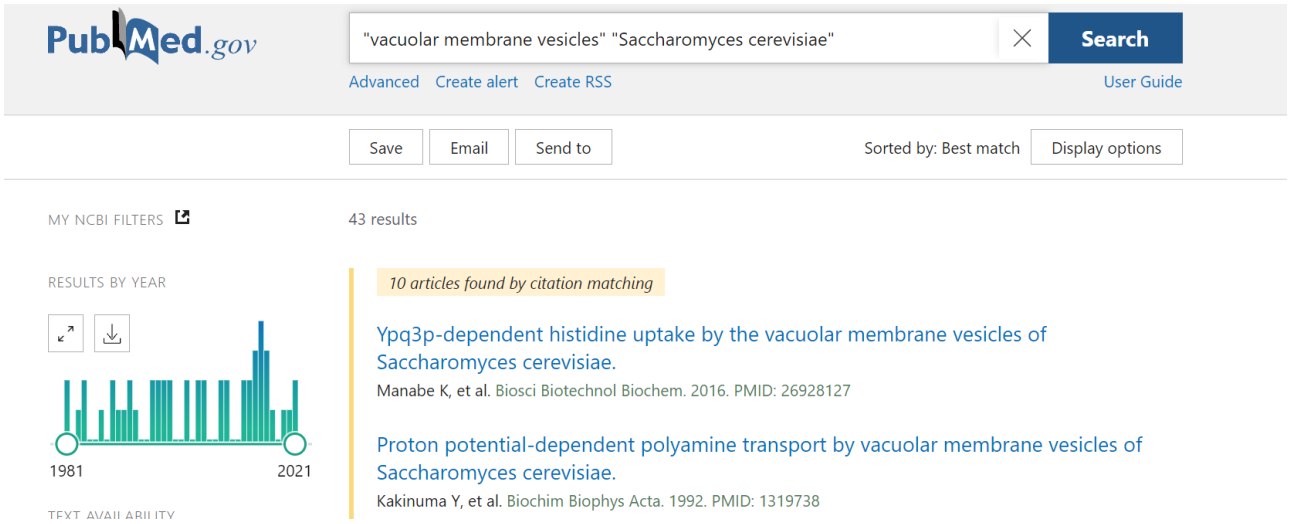
pubmed.gov
NIH National Library of Medicine (National Center for Biotechnology Information)
https://pubmed.ncbi.nlm.nih.gov/

学術論文検索

https://pubmed.ncbi.nlm.nih.gov/により PumMed.gov のページにアクセスしてみましょう。そこで、大隅良典博士が 1981 年に発表した学術論文タイトル、「Active Transport of Basic Amino Acids Driven by a Proton Motive Force in Vacuolar Membrane Vesicles of *Saccharomyces cerevisiae*」の中から、二つのキーワード、液胞膜小胞"vacuolar membrane vesicles"と酵母"Saccharomyces cerevisiae"により学術論文を PubMed で検索してみましょう。

(1) キーワード"vacuolar membrane vesicles"と"Saccharomyces cerevisiae"を入力します。
(2) Search ボタンをクリックします。

検索結果をみると 43 results(43 個)の学術論文を検索することができました。どれでもいいのでタイトルをクリックしてみましょう。すると、選択した論文の要旨が出力されます。この要旨を読んで、いま進めている研究との関係性を考慮し、内容を全て読みたいときには、論文をダウンロードあるいは、図書館などを通してコピーを外注します。そして読み詳細に検討します。金谷・松本はそろそろ 60 歳を迎える研究者ですが、それでも判らない単語がでてくることがあります。その時は、その単語を類推せずに辞書を引きながら読み進めています。目的はしっかり内容を理解するためだからね！



このようにして、学術研究の背景研究を調べて、対象とする研究の新規性と客観性を常に考えながら、新規研究を進め最終的には学術論文にまとめるのが、まあ、研究者のすべきことです。まあ、現状を把握し、どう行動するかという観点では、学術研究の方法自体は、文系、理系の戦略はほぼ同様でしょう。

3. 科学探究は文系の課題解決にも必要です！

科学とは、広義には、一定の目的と方法(科学的方法、思考)をもとに、様々な事象を研究し結論を導く学問です。科学研究は、「仮説検定の学問」とまとめることができます。仮説を立てて検証するというプロセスで研究を進めるからです。科学研究ではデータが伴います。実験であれ、すでに公開されているデータであれ、ともかくデータが必要になります。その意味では、データを収集した時点で科学研究はスタートします。

例えば、高校の近くに二つのハンバーガーレストランができたとしましょう。「20 人の学生さんの中に A 店のハンバーガーが好きだという学生さんが 2 人、残りは B 店が好きだ」という意見が出ました。これは、「A 店と B 店の好みに違いがあるのだろうか？」という仮説へとつながります。そこで、二つのハンバーガー店の選び方をランダムとすると、A 店と B 店を選ぶ確率はそれぞれ 1/2, 1/2 です。

ここで、帰無仮説 H_0「A 店と B 店を選ぶ確率は等しい」という仮説を設定します。では、A 店を 2 人、B 店を 18 人が好きになる確率はどのくらいだろうか？と次に考えます。ここで、高校数学でも習う「二項定理・二項分布」の知識を活用します。

$$\left(0\text{ 人が A 店を選ぶ確率}\right) = {}_{20}C_0\left(\frac{1}{2}\right)^{20} = 1\left(\frac{1}{2}\right)^{20} = 9.536743\cdot 10^{-7}$$

$$\left(1\text{ 人が A 店を選ぶ確率}\right) = {}_{20}C_1\left(\frac{1}{2}\right)^{20} = 10\left(\frac{1}{2}\right)^{20} = 1.907349\cdot 10^{-5}$$

$$\left(2\text{ 人が A 店を選ぶ確率}\right) = {}_{20}C_2\left(\frac{1}{2}\right)^{20} = \frac{10\cdot 9}{2}\left(\frac{1}{2}\right)^{20} = 0.001811981$$

となるので、これらを合計すると A 店を選ぶ人が 2 人以下の確率を求めることができて、

$$\mathrm{P}\left(\text{A 店を選ぶ人数} \leq 2\right) = 0.0002012253$$

となり、通常、統計検定では、閾値 0.05 より小さいとき、H_0 は棄却されて、対立仮説 H_1「B 店を選ぶ確率は A 店より高い」となります。この場合は帰無仮説 H_0「A 店と B 店を選ぶ確率は等しい」が棄却され、対立仮説 H_1「B 店を選択する確率は A 店より統計的に有意に高い」となります。こういう好みの問題について統計学を活用することができます。このように、新規製品の評価とか、ファッションの流行などマーケティングの分野でも統計検定が使われています。

次に、和歌集について桜と梅に興味をもったとします。まず、桜と梅の和歌の数を数えてみましょう（**表 3.1**）。

表 3.1　歌集における桜と梅の和歌の数

歌集 (略号)	万葉集 (Man)	古今和歌集 (Kokin)	後撰集 (Gosen)	金葉集 (Kinyou)	新古今和歌集 (Shinko)	金塊和歌集 (Kinkai)
発行年	759	905	957	1124	1212	1219
和歌の数(桜/梅)	47/119	49/23	35/17	34/9	55/27	57/28

年代を横軸、縦軸に（桜の和歌の数）/（桜と梅の和歌の数）として散布図で表すと**図 3.1** のようになります。いま、ある歌集について、（桜の和歌の数）/（桜と梅の和歌の数）が 0.5 より大きいとき、この歌集では、桜の和歌が梅の和歌よりも多く収録されていることになります。

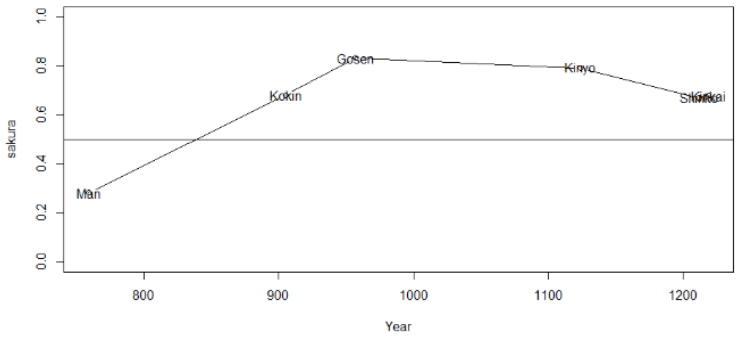


図 3.1 歌集の出版年と（桜の和歌の数/[桜＋梅の和歌の数]）の関係

図 3.1 をみると、確かに万葉集を除く和歌集では、桜の和歌が梅の和歌より多く収録されています。このように、「頻度」に注目すれば文系の課題であっても統計学で客観性を示すことができます。

その上で、「なぜ万葉集（Man）では梅の和歌が桜に比べて多いのだろう？」など疑問がでてきます。これを、歴史的背景、当時の人の好み、桜と梅の植栽状況など多面的に考察して、客観的に説明できれば科学となり得るでしょう。このように、まず客観的事実を定量的に表現すれば、だれもが万葉集の特殊性を直感的に理解できます。その上で、多面的視野で、この現象をどう説明するということになります。これが「理数探究」なのだと思います。ですので、文系・理系に関係なく、まずは自分の興味のある研究をしてみて、さらには「研究した事象を一般化する」あるいは「生活に役立てる方策があるか」などと考えるのもアリでしょう。

「ハンバーガー屋さん A と B の好みに統計的違いはあるか」と「和歌集の桜と梅の和歌の数の変遷」の Python プログラムを Pynarakita13 と Pynarakita14 に示します。実行してみてください！（実際のプログラムの実行は、実習で行います。）

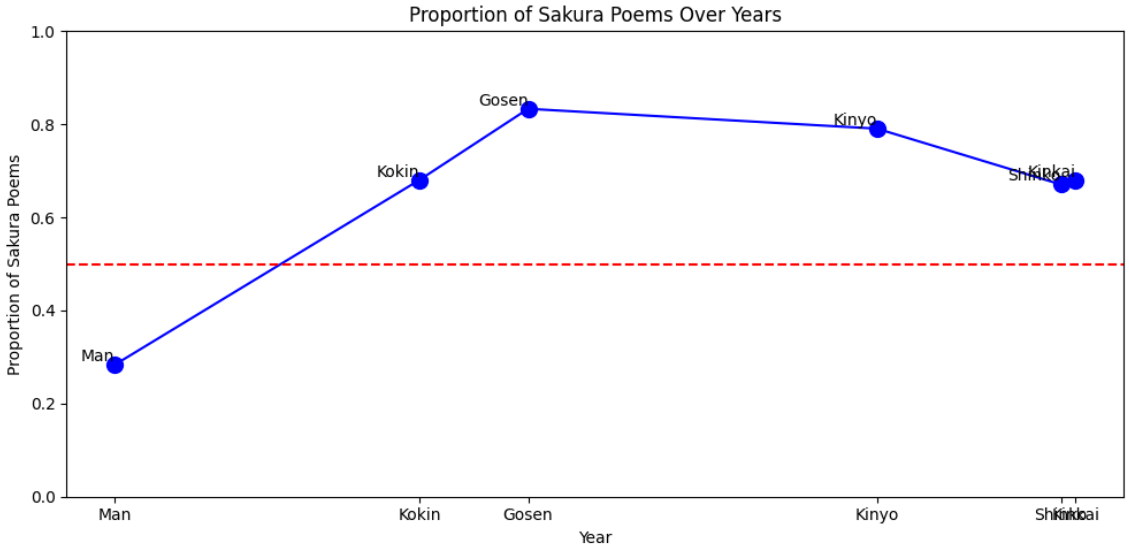
Pynarakita13

```
from scipy.special import comb

total_probability = comb(20, 0) * (1/2)**20 + comb(20, 1) * (1/2)**20 +
comb(20, 2) * (1/2)**20
print(total_probability)
```

Pynarakita14

```
import matplotlib.pyplot as plt

# データを Python のリストに割り当てる
Year = [759, 905, 957, 1124, 1212, 1219]  # 年
sakura = [47/(47+119), 49/(49+23), 35/(35+7), 34/(34+9), 55/(55+27),
57/(57+27)]  # 桜の詩の割合
nameW = ["Man", "Kokin", "Gosen", "Kinyo", "Shinko", "Kinkai"]  # 作品名

# 与えられたデータでプロットを作成する
```

```
plt.figure(figsize=(10, 5))  # プロットのサイズを設定
plt.plot(Year, sakura, marker='o', color='b')  # 線とマーカーでプロットを作成
plt.ylim(0, 1)  # y 軸の範囲を設定
plt.xticks(Year, nameW)  # x 軸にテキストラベルを追加

# 各ポイントにテキストラベルを追加する
for i, txt in enumerate(nameW):
    plt.text(Year[i], sakura[i], txt, verticalalignment='bottom',
horizontalalignment='right')

# y=0.5 で水平線を追加
plt.axhline(y=0.5, color='r', linestyle='--')

# コンテキストのためにラベルとタイトルを追加
plt.xlabel('Year')
plt.ylabel('Proportion of Sakura Poems')
plt.title('Proportion of Sakura Poems Over Years')
plt.tight_layout()

# プロットを表示
plt.show()
```



4. 課題解決法

みなさんは、中学を卒業し、いま中学で学んだことに加え、高校でもさまざまな勉強に励んでいると思います。その中で、「理数探究」という科目をどう活用するといいでしょうか？

この科目は、さまざまな科目を融合させながら、問題を解決することを目指しています。また、専門家の意見も聞きながら実習するというスタイルです。ですので、すでに講義で習ったことを使ってもいいし、習っていないことにチャレンジしても OK です。また、これから習うであろう内容を、先取りして勉強して解くのもいいでしょう。このテキストには、難解な部分もあるかと思います。でも完璧に理解する必要はありません。道具として、「理数探究をどう活用するか」という視点で楽しんでください。

課題を解く方法にはおおよそ二つの方法があります。

(1) 物理法則などの法則から出発して数理モデル式により現象を説明し、この数理による予測、意味づけを行うという方法です。

(2) 実験・調査を通してデータを作成し、そのデータを統計指標で客観的に特徴を理解します。このように一つ一つ事実を積み上げ、一般的法則を導く方法です。

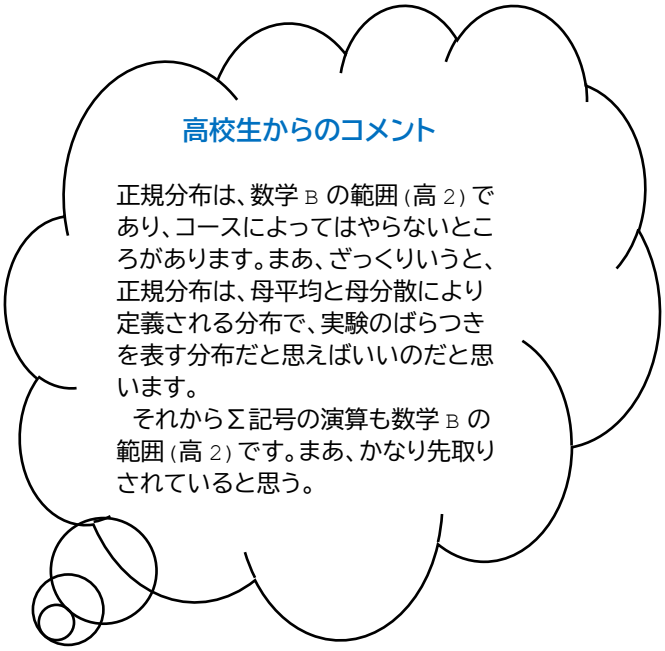
おおまかには、(1)は演繹的方法、(2)は、帰納的方法といいます(啓林館、理数探究基礎:未来に向かって、p.16)。

(1)の場合は数理モデルを導くという作業が必要になり、いろいろな専門書を活用し、数式と格闘することになります。物理・数学大好きな研究者にとっては極めて楽しい作業になります。一方、(2)データを、統計指標をもとに客観的に特徴を理解するのも面白いもので、いままでに考慮しなかった特徴を得ることもできます。ただし(2)は、いまあるデータでの経験的解釈です。あくまで経験的解釈なので、新規法則のてがかりです。というわけでどちらも極めて楽しいのですが、課題を設定したときには答えがありません。まず必要なのは「答え」が未知の問題でもびくびくせずやってみる「ずうずしさ」が必要になります。ここでは、次節では、(2)に関係する統計指標と統計による仮説検定について説明します。

5. 統計検定

5.1 正規分布

データを収集・整理するとそのデータの特徴を理解します。その上で、統計学が必要になります。統計分布の基本は、正規分布です。正規分布を発見したのは、フランスの数学者、ド・モアブル(Abraham de Moivre, 1667-1754)の功績とされています。1730 年代に、ド・モアブルは、二項分布のサンプル数 N を大きくしていくと分布の形が正規曲線で近似できることを発見しました。ガウスは、この分布が非常に好きだったので頻繁に活用しました。それで後世になってガウス分布と呼ばれるようになりました。この分布の一番偉いところは、平均値と分散が定義されると、平均値の区間推定ができるところです。



4 つの値 $x_1 = 10,\ x_2 = 13,\ x_3 = 11,\ x_4 = 10$ について平均値 $\bar{x}$ を求めてみましょう。

$$\bar{x} = \frac{x_1+x_2+x_3+x_4}{4} = \frac{10+13+11+10}{4} = 11$$

ここで、$x_1 + x_2 + x_3 + x_4$ をΣ記号を用いて、$\sum_{i=1}^{4} x_i$ と書きます。ここで、$\sum_{i=1}^{4}$ とは、$i = 1$ から 4 まで合計するという意味です。そこで、x_i となっているので、$\sum_{i=1}^{4} x_i$ は $x_1 + x_2 + x_3 + x_4$ となります。このように、Σ記号を使えば、$\bar{x} = \frac{1}{4}\sum_{i=1}^{4} x_i$ と短く書くことができます。

つづいて分散を求めてみましょう。N 個のサンプル $x_1, x_2, \cdots, x_N$ それぞれついて平均値の差の 2 乗を計算します。これら N 個の差の 2 乗の合計を、[サンプル数]-1 で割った値を不偏分散 U^2 と呼びます。Σ記号使って定義し、計算すると以下のようになります。

$$U^2 = \frac{\sum_{i=1}^{N}(x_i-\bar{x})^2}{N-1} = \frac{(10-11)^2+(13-11)^2+(11-11)^2+(11-10)^2}{4-1} = \frac{1^2+2^2+0^2+1^2}{3} = \frac{1^2+2^2+0^2+1^2}{3} = 2$$

なお、高校数学の教科書では、分散を以下の式で定義します。ここで分散の式の分母はサンプル数 N です。

$$U^2 = \frac{\sum_{i=1}^{N}(x_i-\bar{x})^2}{N}$$

このテキストでは不偏分散を主に説明します。大学で不偏分散と分散の違いを習います。「不偏分散とはサンプル数が影響を与えない分散と理解しておきましょう。」

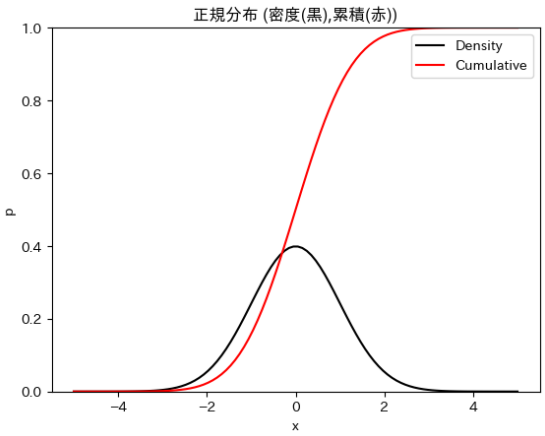
Pynarakita15 を実行し、平均(mean)0、標準偏差(sd)1 の正規分布を描画してみましょう。

Pynarakita15

```
import japanize_matplotlib
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import norm

xmin = -5  # x 軸の最小値
xmax = 5   # x 軸の最大値

# 正規分布の確率密度関数をプロット
x_values = np.linspace(xmin, xmax, 100)  # x 値を生成
y_values = norm.pdf(x_values, 0, 1)      # 平均 0、標準偏差 1 の正規分布の確率密度関数

plt.plot(x_values, y_values, color='black', label='Density')  # 密度関数を黒色でプロット

# 正規分布の累積分布関数をプロット
y_values_cumulative = norm.cdf(x_values, 0, 1)  # 累積分布関数
plt.plot(x_values, y_values_cumulative, color='red', label='Cumulative')
# 累積分布関数を赤色でプロット

# プロットのタイトルと軸ラベルを設定
plt.title('正規分布 (密度(黒),累積(赤))')
plt.xlabel('x')
plt.ylabel('p')
plt.ylim(0, 1)  # y 軸の範囲を 0 から 1 に設定
plt.legend()    # 凡例を表示

# プロットを表示
plt.show()
```

図の黒の分布が正規分布の密度分布です。$-\infty$ から $+\infty$ まで足し算すると 1 になります(確率分布の総和は 1 となります)。一方、赤は累積分布で、$-\infty$ から x までの確率値を表します。これをみると、-4 ではほぼ 0 ですが、$+4$ になるとほぼ 1 になります。

また、例えば $-\infty$ から平均値 0 までの確率は $P(-\infty < x \leq 0) = 0.5$、平均値 0 から $+\infty$ までの確率は $P(0 < x < \infty) = 0.5$ となります。まあ、平均値 0 の右と左は対称である分布なので納得いくか、と思います。



いま、母平均と母分散が既知であれば、母平均 μ と実際にデータから得られた平均値 $\bar{x}$ と母分散 σ^2 からなる $\frac{\bar{x}-\mu}{\sqrt{\frac{\sigma^2}{N}}}$ は平均 0 分散 1 の正規分布に従います。ここで N はサンプル数を表しています。そこで、$p = 0.025$ と $p = 0.975$ の区間の累積確率 $P = 0.975 - 0.025 = 0.95$ となるので、$\frac{\bar{x}-\mu}{\sqrt{\frac{\sigma^2}{N}}}$ の 95%信頼区間 (同様の分布から 100 回平均値を求めたときに 95%の平均値が含まれる範囲) となります。$p = 0.025$ と $p = 0.975$ の正規分布の x の値がわかれば、$\frac{\bar{x}-\mu}{\sqrt{\frac{\sigma^2}{N}}}$ の区間を推定することができます。

これは、

```
norm.cdf(0.025, 0, 1)= -1.959964
```

```
norm.cdf(0.975, 0, 1)= 1.959964
```

と Python により計算することができます。

$$-1.959964 < \frac{\bar{x}-\mu}{\sqrt{\frac{\sigma^2}{N}}} < 1.959964$$

となり、これを変形すると、

$$\bar{x} - 1.959964\sqrt{\frac{\sigma^2}{N}} < \mu < \bar{x} + 1.959964\sqrt{\frac{\sigma^2}{N}}$$

となり、母平均 μ の区間推定となります。サンプル数 N が大きい時は、実際のサンプルの分散を σ^2 として計算することができます。ですが、母分散など本来わかりません。そこで、正規分布での区間推定はあまり使われません(ただ正規分布からさまざまな分布が誘導されて統計学ができあがっているのでとりあえず必要です)。それにとってかわったのが t 分布です。

5.2 t 分布

正規分布では母分散が必要なのですが、これにサンプル数を考慮することにより、母分散を活用せずに母平均の区間推定ができるのが t 分布です。t 分布の発明者ウィリアム・シーリー・ゴセット博士 (William Sealy Gosset) は、1899 年にギネスビール社のダブリン醸造所に就職し、統計学の知識で醸造と農業(オオムギの改良)の研究をしていました。ギネスビール社では企業秘密の問題で社員が論文を出すことを禁止していたので、ゴセットは Student (スチューデント) というペンネームで論文を発表したのです。それで一般にはスチューデントの t 分布と呼ばれています。

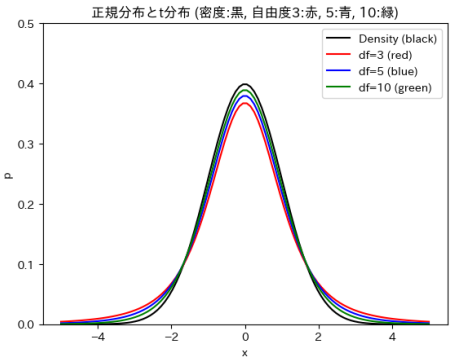
t 分布では[サンプル数-1]を自由度といいます。この自由度で学生さんたちは、その意味をまじめに考えすぎて、混乱します。金谷も学生の頃、自由度について調べすぎて混乱したことがあります。なにせに「熱力学」「力学」にも自由度という用語が出てきます。が、ここは「統計学の自由度です」。まあ適にスルーしてください。憶えておくことはサンプル数 N により分布が異なるということです。「t 分布では $N-1$ が自由度だよ！」と覚えてしまうのがいいでしょう。

Pynarakita16 を実行して t 分布をみてみましょう。

Pynarakita16

```
import japanize_matplotlib
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import norm

xmin = -5  # x 軸の最小値
xmax = 5   # x 軸の最大値

# x の値を生成
x = np.linspace(xmin, xmax, 100)

# 正規分布の密度関数をプロット
plt.plot(x, norm.pdf(x, 0, 1), color='black', label='Density (black)')

# 自由度 3, 5, 10 の t 分布をプロット
plt.plot(x, t.pdf(x, 3), color='red', label='df=3 (red)')
plt.plot(x, t.pdf(x, 5), color='blue', label='df=5 (blue)')
plt.plot(x, t.pdf(x, 10), color='green', label='df=10 (green)')

# プロットの設定
plt.title('正規分布と t 分布 (密度:黒, 自由度 3:赤, 5:青, 10:緑)')
plt.xlabel('x')
plt.ylabel('p')
plt.ylim(0, 0.5)  # y 軸の範囲を設定
plt.legend()      # 凡例を表示

# プロットを表示
plt.show()
```

図は Pynarakita16 の実行により得られる t 分布です。正規分布を黒線で表しました。赤はサンプル数 4（自由度 3）、青はサンプル数 6（自由度 5）、緑はサンプル数 11（自由度 10）のときの分布です。このようにサンプル数を増やしていくと、正規分布に近づく分布が t 分布です。



t 分布を用いて、平均値の区間推定をしてみましょう。

サンプル数 N におけるサンプルについての平均値 $\bar{x}$ と真の平均値 μ の差の 95%信頼区間は以下の式により表現できます。

$$\mathrm{t}\left(p=0.025, \text{自由度}=N-1\right) < \frac{\bar{x}-\mu}{\sqrt{\frac{U^2}{N}}} < \mathrm{t}\left(p=0.975, \text{自由度}=N-1\right)$$

ここで、$U^2 = \frac{\sum_{i=1}^{N}(x_1-\bar{x})^2}{N-1}$（分散）です。正規分布のときは母分散でしたが、t 検定ではここが標本分散、すなわちデータから計算できる分散です。
では、上の式で、

$$\mathrm{t}\left(p=0.025, \text{自由度}=N-1\right) = -\mathrm{Thr}、\ \mathrm{t}\left(p=0.975, \text{自由度}=N-1\right) = \mathrm{Thr}$$

とおき、真の平均値 μ の区間を求めると、

$$\bar{x} - \mathrm{Thr}\sqrt{\frac{U^2}{N}} < \mu < \bar{x} + \mathrm{Thr}\sqrt{\frac{U^2}{N}}$$

となります。

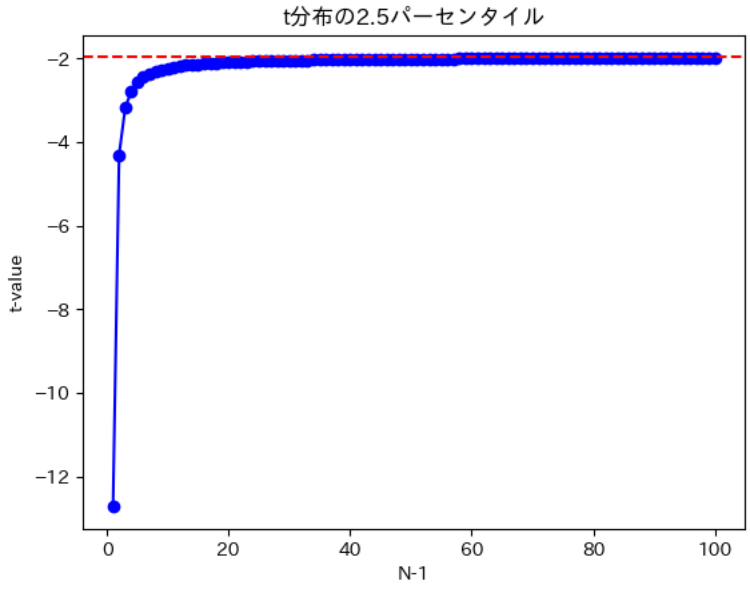
Python を用いて、$\mathrm{t}\left(p=0.025, \text{自由度}=N-1\right)$を計算してみましょう。Pynarakita23 では自由度を 1-100 まで設定し計算しました。

Pynarakita23

```
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import norm, t

ns = 100  # サンプルサイズ
qt25 = [t.ppf(0.025, df) for df in range(1, ns + 1)]  # t 分布の 2.5 パーセンタイルを計算

# t 分布の 2.5 パーセンタイルをプロット
plt.plot(range(1, ns + 1), qt25, marker='o', linestyle='-', color='blue')
# type="b"は青色の線とマーカー
plt.xlabel('N-1')  # x 軸のラベル
plt.ylabel('t-value')  # y 軸のラベル
```

```
plt.title('t 分布の 2.5 パーセンタイル')

# 正規分布の 2.5 パーセンタイルを計算 (p=0.025)
qnorm_0025 = norm.ppf(0.025, 0, 1)

# 結果を表示
plt.axhline(y=qnorm_0025, color='red', linestyle='--')  # 正規分布の 2.5 パーセンタイルを赤い点線で追加
plt.show()

print(qnorm_0025)
```

Pynarakita23 により得られた図を見てみましょう。



この図をみると $p = 0.025$ について、$N - 1$ が $1, 2, \cdots, 100$ とすると
`-12.706205, -4.302653, -3.182446,…,`
`-1.984467, -1.984217,-1.983972` となります。これらの値の絶対値が

$$\bar{x} - \mathrm{Thr}\sqrt{\frac{U^2}{N}} < \mu < \bar{x} + \mathrm{Thr}\sqrt{\frac{U^2}{N}}$$

の `Thr` に対応します。いま、データの平均値 $\bar{x}$ と分散 U^2 が同じであったとしても、サンプル数と対応した値 `Thr` を定義するというのが `t` 分布の特徴です。ちなみに、正規分布の `Thr` は常に `1.959964` となります。

(a) 1 標本の母平均 μ が μ_0 とみなせるか検定する

帰無仮説 H_0 : $\mu = \mu_0$

対立仮説 H_1： $\mu \neq \mu_0$

とします。統計検定では、仮説として、母平均 μ は μ_0 と等しいという仮説を立てます。この仮説を帰無仮説 H_0 と呼びます。帰無仮説が起こる確率 p を求めます。この確率 p が非常に小さい場合、帰無仮説を棄却して、対立仮説 H_1 を支持します。すなわち、対立仮説 $H_1 : \mu \neq \mu_0$ を採択するために、帰無仮説 H_0 の起こる確率が小さい（おおまかには $p < 0.05$ ）ことを示します。

いま、U^2 を用いると、$\frac{\bar{x}-\mu_0}{\sqrt{\frac{U^2}{N}}}$ は、自由度$N-1$の t 分布に従います。そこで、

$p\left(\mathrm{t} = \frac{\bar{x}-\mu_0}{\sqrt{\frac{U^2}{N}}}, df = N-1\right) < 0.05$ のとき、帰無仮説 $H_0 : \mu = \mu_0$ が棄却されて、対立仮説 $H_1 : \mu \neq \mu_0$ が採択されます。ここで、μ は実測値 x の平均値 $\bar{x}$ の母平均を表します（ちょっとややこしい）。

ヒトの時間予測の精度が学習回数によりどこまで向上するか？という疑問を解明するために、ストップウオッチなどを使って、ちょうど 5 秒だと思ったときに止め、時間を確認するという実験をしました。この操作を繰り返すことにより、以下の結果を得ました。このとき、10 回の実験の時間の平均値は 5 秒とみなせるだろうか。

回数	1	2	3	4	5	6	7	8	9	10
時間（秒）	5.26	4.78	5.09	4.57	5.06	5.07	4.96	5.10	4.84	4.97

［啓林館、理数探究基礎：未来に向かって p.42, p.125］

この問題を Pynarakita17 で解いてみましょう。x には 10 回の実験の測定値が格納されている。これらの平均値は、5 秒を予測して実験しているので

$\mu_0 = 5$

とします。すなわち、

帰無仮説 $H_0 : \mu = 5$ ， 対立仮説 $H_1 : \mu \neq 5$

とします。

Pynarakita17

```
import numpy as np
from scipy.stats import ttest_1samp, t

x = [5.26, 4.78, 5.09, 4.57, 5.06, 5.07, 4.96, 5.10, 4.84, 4.97]  # サンプルデータ

# 片側 t 検定を実施
t_statistic, p_value = ttest_1samp(x, popmean=5)

# 結果を出力
```

```
print("t:", t_statistic)
print("p 値:", p_value / 2)  # 片側検定のため、p 値を 2 で割る

# 標本平均と標準誤差
sample_mean = np.mean(x)
print("サンプル平均:", sample_mean)
sample_std = np.std(x, ddof=1)
se = sample_std / np.sqrt(len(x))

# 自由度
df = len(x) - 1

# t 分布を用いて 95%信頼区間を計算
confidence_level = 0.95
t_critical = t.ppf((1 + confidence_level) / 2, df)
confidence_interval = (sample_mean - t_critical * se, sample_mean +
t_critical * se)

print("95%信頼区間:",confidence_interval)
```

Pynarakita17 を実行すると、

```
t: -0.483
p 値: 0.320
サンプル平均: 4.97
信頼区間: (4.83, 5.11)
```

となり、$p < 0.05$ でないので、帰無仮説 H_0 ： $\mu = 5$ を棄却することができない。つまり、実験 10 回のデータの平均値 μ は 5 とみなすことができました。ちなみに、μ の 95%信頼区間は、

$$4.83 < \mu < 5.11$$

という範囲で、5 を含んでいます。

(b) 2 標本問題の母平均の差を検定する

二つの標本の母分散が等しいと仮定できないとき、

$t = \frac{\bar{x}-\bar{y}}{\sqrt{\frac{U_x{}^2}{N_x}+\frac{U_y{}^2}{N_y}}}$ は、自由度 $f = \frac{\frac{U_x{}^2}{N_x}+\frac{U_y{}^2}{N_y}}{\frac{U_x{}^2}{N_x(N_x-1)}+\frac{U_y{}^2}{N_y(N_x-1)}}$ の t 分布に従います。

なにやら難しそうな式ですが、t.test()関数では、データを入力すれば自動的に計算してくれます。

二つの sampleD と sampleL について 10 回の測定を行った。この二つの sampleD と sampleL の平均値に有意な差があるだろうか。
sampleD={0.7,-1.6,-0.2,-1.2,-0.1, 3.4,3.7,0.8,0.0,2.0}
sampleL={1.9, 0.8, 1.1, 0.1,-0.1, 4.4,5.5,1.6,4.6,3.4}

いま、帰無仮説 $H_0 : \bar{\mu}_{\mathrm{sampleD}} = \bar{\mu}_{\mathrm{sampleL}}$ とします。帰無仮説 $H_1 : \bar{\mu}_{\mathrm{sampleD}} \neq \bar{\mu}_{\mathrm{sampleL}}$ となります。

Pynarakita18

```
from scipy.stats import ttest_ind

# サンプルデータを定義します。
sampleD = [0.7, -1.6, -0.2, -1.2, -0.1, 3.4, 3.7, 0.8, 0.0, 2.0]
sampleL = [1.9, 0.8, 1.1, 0.1, -0.1, 4.4, 5.5, 1.6, 4.6, 3.4]

# 二標本検定を行います。対応がなく、等分散ではないと仮定しています。
result = ttest_ind(sampleD, sampleL, alternative='two-sided',
equal_var=False)

print(result)
```

Pynarakita18 を実行すると、$p = 0.079$ となり $p < 0.05$ とならないので、sampleD と sampleL の帰無仮説 $H_0 : \bar{\mu}_{\mathrm{sampleD}} = \bar{\mu}_{\mathrm{sampleL}}$ を棄却できないので、sampleD と sampleL の平均値は同じであると判断されます。

TtestResult(statistic=-1.86, pvalue=0.0794, df=17.8)

について、equal_var.equal=False としています。equal_var.equal=False は、二つの標本の分散は異なるとして検定をしています。ここで、二つの標本の分散が同じであるということはまずありません。それで通常は equal_var.equal=False として検定します。

続いて、例えば被験者に対応がとれるときにの場合について考えましょう。いま、10 人の被験者 A-J について sampleD と sampleL の薬を呑んだ時、通常の睡眠時間に比べて増加したか減少したかについて測定したとしましょう。この場合、sampleD と sampleL の値はそれぞれの被験者について測定されているので、対応のあるデータの t 検定により検定できます。

	A	B	C	D	E	F	G	H	I	J
sampleD	0.7	-1.6	-0.2	-1.2	-0.1	3.4	3.7	0.8	0.0	2.0
sampleL	1.9	0.8	1.1	0.1	-0.1	4.4	5.5	1.6	4.6	3.4

この場合、sampleD と sampleL の測定値について、それぞれの被験者について差分を取ることができます。つまり、H_0：「被験者 A-J の差分の平均値が 0 とみなせる」として t 検定をすることもできます。Pynarakita19 を実行してみましょう。

Pynarakita19

```
import numpy as np
from scipy.stats import ttest_rel, ttest_1samp

# サンプルデータを numpy 配列として定義
sampleD = np.array([0.7, -1.6, -0.2, -1.2, -0.1, 3.4, 3.7, 0.8, 0.0, 2.0])
sampleL = np.array([1.9, 0.8, 1.1, 0.1, -0.1, 4.4, 5.5, 1.6, 4.6, 3.4])

# [1] 対応のある二標本検定を行う。
result_paired = ttest_rel(sampleD, sampleL)

# [2] sampleD と sampleL の差分を計算し、その差分に対して一標本検定を行います（母平均が 0 と異なるかの検定）。
dif = sampleD - sampleL
result_one_sample = ttest_1samp(dif, popmean=0)

print(result_paired)
print(result_one_sample)
```

[1]では、対応のある二標本検定とした場合の t 検定の結果です。一方、[2]では、sampleD-sampleL を計算しその平均値が0とみなせる（H_0）ということを検定した場合です。これらの結果を見るといずれも同様の結果（p-value = 0.00283） が得られます。つまり、対応のある二標本検定とした 2 群の検定も sampleD-sampleL の平均値が 0 とみなす検定と同じであることがわかると思います。データについて対応が取れる場合の t 検定では、p < 0.05 となり、二つのサンプル sampleD と sampleL は統計的に有意となりました。

このように、測定データについて対を成している場合となしていない場合で、使う統計検定法が異なりますので、データに合った統計検定法を選ぶましょう。

```
> #[1]
TtestResult(statistic=-4.06, pvalue=0.00283, df=9)

> #[2]
TtestResult(statistic=-4.06, pvalue=0.00283, df=9)
```

5.3　χ2 乗分布

	B_1	B_2	…	B_m	合計
A_1	O_{11}	O_{12}	…	O_{1m}	R_1
A_2	O_{21}	O_{22}	…	O_{2m}	R_2
…	…	…	…	…	…
A_l	O_{l1}	O_{l2}	…	O_{lm}	R_l
合計	C_1	C_2		C_m	N

分割表（クロス集計）とは、2 つの項目 A の l 個の変数（A_1 – A_l）と項目 B の m 個の変数（B_1 – B_m）のサンプルの数 O_{ij} からなる表のことをいいます。

（a）　χ2 独立性の検定

要因 A と B が独立だとすると、A_i, B_j の期待値 E_{ij} は

$$E_{ij} = \frac{R_i C_j}{N}$$

となります。そこで、カイ2乗値を以下のように定義します。

$$\chi^2 = \sum_{j=1}^{m} \sum_{i=1}^{l} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

このカイ二乗値（χ^2）、は自由度 $\phi = (l-1)(m-1)$ のもとで χ^2 分布に従います。そこで、

$\chi^2 \leq \chi_\alpha^2$ のとき「独立である（要因 A と B は関係ない）」。一方、

$\chi^2 > \chi_\alpha^2$ のとき「独立でない（要因 A と B は関係ある）」

となります。ここで、有意水準 α における基準カイ乗値を χ_α^2 で表します。通常 $\alpha = 0.05$ とします。

2x2 分割表（クロス集計）

実験データ		--------- 要因B ---------		
		B_1	**B_2**	**Total**
要因A	**A_1**	a	b	$R_1 = a + b$
	A_2	c	d	$R_2 = c + d$
	Total	$C_1 = a + c$	$C_2 = b + d$	$N = a + b + c + d$

周辺確率から予測される頻度		--------- 要因B ---------		
		B_1	**B_2**	**Total**
要因A	**A_1**	$N(A_1 \cap B_1)$	$N(A_1 \cap B_2)$	$R_1 = a + b$
	A_2	$N(A_2 \cap B_1)$	$N(A_2 \cap B_2)$	$R_2 = c + d$
	Total	$C_1 = a + c$	$C_2 = b + d$	$N = a + b + c + d$

2x2 分割表について、4 つの観測値を a, b, c, d とします(上段)、一方、周辺確率から予測される頻度を, $N(A_1 \cap B_1)$, $N(A_1 \cap B_2)$, $N(A_2 \cap B_1)$, $N(A_2 \cap B_2)$ とすると、カイ2乗値は、

$$\chi^2 = \frac{\{a-N(A_1 \cap B_1)\}^2}{N(A_1 \cap B_1)} + \frac{\{b-N(A_1 \cap B_2)\}^2}{N(A_1 \cap B_2)} + \frac{\{c-N(A_2 \cap B_1)\}^2}{N(A_2 \cap B_1)} + \frac{\{d-N(A_2 \cap B_2)\}^2}{N(A_2 \cap B_2)} \qquad (1)$$

となります。この式を a, b, c, d により表してみましょう。

周辺確率を a, b, c, d で表すと以下のようになります。

$$N(A_1 \cap B_1) = N \cdot P(A_1 \cap B_1) = N \cdot P(A_1) \cdot P(B_1) = \frac{(a+c)(a+b)}{N}$$

$$N(A_1 \cap B_2) = N \cdot P(A_1 \cap B_2) = N \cdot P(A_1) \cdot P(B_2) = \frac{(a+b)(b+d)}{N}$$

$$N(A_2 \cap B_1) = N \cdot P(A_2 \cap B_1) = N \cdot P(A_2) \cdot P(B_1) = \frac{(c+d)(a+c)}{N}$$

$$N(A_2 \cap B_2) = N \cdot P(A_2 \cap B_2) = N \cdot P(A_2) \cdot P(B_2) = \frac{(b+d)(c+d)}{N}$$

これらを (1) に代入すると、

$$\chi^2 = \frac{(ad-bc)^2 N}{(a+b)(c+d)(a+c)(b+d)}$$

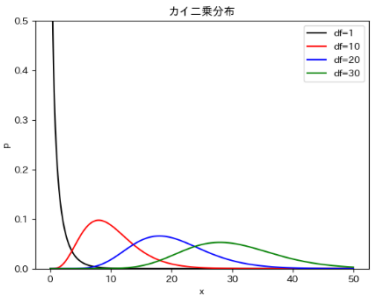
と簡単になり、この時の自由度 $\phi = (2-1)(2-1) = 1$ となります。

χ^2 分布を Pynarakita20 によりみてみましょう。

Pynarakita20

```
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import chi2

# x の最小値と最大値を設定
xmin = 0
xmax = 50

# x の値の範囲を作成
x = np.linspace(xmin, xmax, 1000)

# カイ二乗分布の確率密度関数をプロットします。自由度が 1 の場合(黒色)
plt.plot(x, chi2.pdf(x, df=1), color="black", label='df=1')

# 自由度が 10 の場合(赤色)
plt.plot(x, chi2.pdf(x, df=10), color="red", label='df=10')

# 自由度が 20 の場合(青色)
plt.plot(x, chi2.pdf(x, df=20), color="blue", label='df=20')
```

```
# 自由度が 30 の場合（緑色）
plt.plot(x, chi2.pdf(x, df=30), color="green", label='df=30')

# グラフのタイトルと軸ラベルを設定
plt.title('カイ二乗分布')
plt.xlabel('x')
plt.ylabel('p')
plt.ylim(0, 0.5)  # y 軸の範囲を設定

# 凡例を表示
plt.legend()

# グラフを表示
plt.show()
```



Pynarakita20 により得られる関数をみると、黒色は自由度 1、赤は自由度 10、青は自由度 20、緑は自由度 30 のときの関数の概形です。このように分割表のサイズにより活用する χ^2 分布は異なります。

コレラと水道会社は独立（関係ない）か？

ジョン・スノウ（John Snow）は、英国の医師で、麻酔と医療衛生の発展における先導者です。1854 年にロンドンで起きたコレラの発生原因を追跡しました。当時、まだコレラという細菌は同定されていませんでした。とりあえず、関係がありそうな要因を整理しました。水道会社（A と B）とコレラの死亡者の有無という二つの因子について整理した結果を以下に示します。

(A)	コレラによる死亡者が出た世帯数	コレラによる死亡者が出ていない世帯数
水道会社A	1263	38783
水道会社B	98	26009

帰無仮説 H_0： 水道会社とコレラの死亡者の有無の出現数は独立である（関係がない）
帰無仮説 H_1： 水道会社とコレラの死亡者の有無の出現数は独立でない（関係がある）
とします。

Pynarakita21

```
import pandas as pd
from scipy.stats import chi2_contingency

# コレラデータと非コレラデータを定義
cholera = [1263, 98]
```

```
noncholera = [38783, 26009]

# データフレームを作成
crossdataA = pd.DataFrame({
    'cholera': cholera,
    'noncholera': noncholera
})

# データフレームの表示
print(crossdataA)

# カイ二乗検定を実行し、結果を表示
chi2, p, dof, expected = chi2_contingency(crossdataA)

# 結果を出力
print('chi2:', chi2)
print('p:', p)
print('自由度:', dof)
print('Expected frequencies:', expected)
```

Pynarakita21 では、crossdataA として、コレラ死亡の有無による世帯数と、水道局 A、B の使用数の関係を示した 2 x 2 のクロス表を作成しました。

```
> print(crossdataA)
   cholera  noncholera
0     1263       38783
1       98       26009
```

crossdataA をもとに chi2_contingency を実行すると p-value < 2.2e-16 となり、コレラ死亡の有無による世帯数と、水道局 A、B の使用数の間は独立ではない（すなわち関係がある）。

```
> chi2_contingency(crossdataA)
chi2: 604.1095893424275
p: 2.137645522374376e-133
自由度: 1
```

次の B-D について二つの因子は独立でしょうか？カイ二乗検定を行い検討してみましょう。

(B)	売れた数	在庫数
既存の製品	9500	90500
新規デザイン	9600	90400

(C)男性	喫煙者	非喫煙者
肺がん患者	1350	7
非肺がん患者	1296	61

(D)女性	喫煙者	非喫煙者
肺がん患者	68	40
非肺がん患者	49	59

(b) χ^2 適合度の検定

帰無仮説 H_0 と対立仮説 H_1 を以下のように設定します。

H_0 :観測度数と期待度数は等しい(理論と実験の結果は適合している)

H_1 :観測度数と期待度数は等しくない(理論と実験の結果は適合していない)

母集団が M 個の排他的事象 $A_1,\ A_2,\cdots,A_M$ に分けられていて、それぞれの観測度数を $c_1,\ c_2,\cdots,c_M$ としよう($c_1+c_2+\cdots+c_M=N$)。またそれぞれの事象の理論確率 $p_1,p_2,\cdots,p_M$ とします。

事象	A_1	A_2	$\cdots$	A_i	$\cdots$	A_M
観測度数	c_1	c_2	$\cdots$	c_i	$\cdots$	c_M
理論度数	Np_1	Np_2	$\cdots$	Np_i	$\cdots$	Np_M

観測地と理論度数により χ^2 を計算します。

$$\chi^2=\sum_{i=1}^{M}\frac{(c_i-Np_i)^2}{Np_i}$$

この χ^2 は、自由度 $k-1$ の χ^2 分布に従います。いま、

$$\chi^2\geq\chi^2\ (\text{自由度}=k-1,\ p=0.05)$$

のとき、H_0 が棄却されます。また、

$$\chi^2<\chi^2(\text{自由度}=k-1,\ p=0.05)$$

のとき、H_0 が採択されます。適合度の検定では、メンデルの実験で有名です。遺伝学者メンデルは、エンドウマメの黄種、緑種の数を数えたら、それぞれ、6022、2001 個となりました。遺伝の法則に従うと 3:1 の分離比が成り立っているでしょうか？以下のプログラムを実行してみましょう。

Pynarakita22

```
from scipy.stats import chisquare

# 実験データと理論比率を定義
experiment = [6022, 2001]
theoretical = [3/4, 1/4]

# カイ二乗検定を実行し、結果を表示
# 実験データに対する理論比率に基づいて期待度数を計算し、カイ二乗統計量と p 値を求めます。
```

```
chi2_stat, p_val = chisquare(experiment, f_exp=[sum(experiment) *
prop for prop in theoretical])

# 結果を出力
print('chi2:', chi2_stat)
print('p:', p_val)
```

chisquare の中の x が実測値 6022、2001 であり、p が理論確率 3/4, 1/4 となっています。理論確率は総和が 1 になるように定義します。結果をみてみましょう。$\chi^2 = 0.014999$ と極めて小さく、また $p > 0.05$ なので、H_0 が採択されます。これにより、実測値 6022、2001 は理論確率 3/4、1/4 とみなしてよいことがわかりました。

```
> chisquare(experiment, f_exp=[sum(experiment) * prop for prop in
theoretical])
chi2: 0.01499854584735552
p: 0.9025279515853211
```

このように統計検定を活用すると、例えば平均値などの統計量を通して、有意な差があるかどうかを確率値 p により評価することができます。

高校生からのコメント（思考実験）

実際に「統計検定」を使う、最も高い可能性があるのは、おそらく、「マーケティング調査」であろう。これを実証実験として、例えば、学園祭で、マーケティング調査実験をしてはどうだろうか。いろいろな商店を調査し、来店人数、単品ごとの価格、売り上げ数を調査する。売上数を y として、来店人数と単品ごとの価格から売上数を求める回帰モデルとつくり、このモデルをもとに学園祭当日の来訪者と価格からの売上数をそれぞれ商品について予測する。その上で、学園祭でこの予測値にどのくらいあうかを検討してみるという計画を漠然とではあるが、思いついた。このような発想で、商品の仕入れにおける最適解を考えるのも、実証実験として面白そうだと思う。

6. ひたすらプログラムをつくり、解析しよう！

「理数探究基礎：未来に向かって」では、さまざまな数値例が説明されています。これらの数値例を出来る限り具体的に解析し、「理数探究」としていろいろ考察してみましょう。また、これらの数値例の解析を通して、解析ができる新規課題を提案してみましょう。

6.1 統計検定を活用して仮説を検定する

教科「理数」で行う「探究」では、自然科学の手法でさまざまな現象を説明することが求められます。さらに、説明できた現象をもとにした機械などを考案できるとなおかっこいい！現象を説明するために活用される自然科学の手法として統計学（統計検定）の基礎についてはすでに 5 節で学びました。そこで、教科書の例をもとに、実際に統計検定を通して仮説の検証を行ってみましょう。

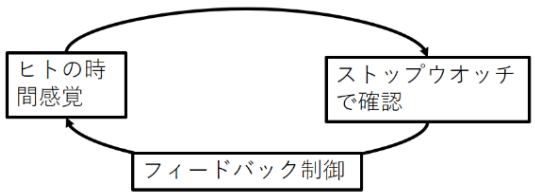
ヒトの時間予測の精度が学習回数によりどこまで向上するか？という疑問を解明するために、ストップウオッチなどを使って、ちょうど 5 秒だと思ったときに止め、時間を確認するという実験をしました。この操作を繰り返すことにより、以下の結果を得ました。

回数	1	2	3	4	5	6	7	8	9	10
時間（秒）	5.26	4.78	5.09	4.57	5.06	5.07	4.96	5.10	4.84	4.97

［啓林館、理数探究基礎：未来に向かって p.42, p.125］

この課題の仮説を考えてみましょう。

「被験者の予測時間の平均値が 5 秒とみなせるか？」については、5 節（b1）で実習しました。そこでは、5 秒とみなすことができました。さらに、「5 秒を予測する。その後、ストップウオッチで時間を確認する。これを繰返す。」という実験だから、毎回、感覚時間をストップウオッチで確認することにより、被験者の時間感覚を修正し、時間の予測精度が向上する（5 秒に近づく）と期待できます。フィードバック制御により被験者の時間感覚の精度が向上すると考えられます。



平均値

10 回の測定時間の和を測定回数で割ると平均値が計算できます。

$$\left(\text{平均値}\right) = \frac{5.26+4.78+\cdots+4.97}{10}$$

1 回目の測定値を $x_1 = 5.26$、2 回目の測定値を $x_2 = 4.78$ と、i 回目の測定値を x_i と表記すると、平均値 $\bar{x}$ は、

$$\bar{x} = \frac{x_1+x_2+\cdots+x_{10}}{10} = \frac{\sum_{i=1}^{N} x_i}{N}$$

と定義できます。ここで N は測定回数を表しています。

$\sum_{i=1}^{N} x_i$ とは、N 個の x_i $(i = 1,2,\cdots,N)$ の合計を表しています。そこで、1 回目から 10 回目の平均値 (zentai)、はじめの4回の平均値 (hajime)、後ろの4回の平均値 (ushiro) を計算するプログラムを作成してみましょう。

Pynarakita03

```
import numpy as np

# データの定義
x = np.array([5.26, 4.78, 5.09, 4.57, 5.06, 5.07, 4.96, 5.10, 4.84, 4.97])

# [1] 各種平均を計算
# 全体の平均
zentai = np.sum(x) / 10
# 最初の 4 つの平均
hajime4 = np.sum(x[:4]) / 4
# 最後の 4 つの平均
ushiro4 = np.sum(x[6:10]) / 4

# [2] numpy の mean 関数を使った平均の計算
# 全体の平均
mean_zentai = np.mean(x)
# 最初の 4 つの平均
mean_hajime4 = np.mean(x[:4])
# 最後の 4 つの平均
mean_ushiro4 = np.mean(x[6:10])

# [3] 奇数番目と偶数番目の平均の計算
# 奇数番目の平均
kisu = np.sum(x[[0, 2, 4, 6, 8]]) / 5
# 偶数番目の平均
gusu = np.sum(x[[1, 3, 5, 7, 9]]) / 5

# 結果の出力
print('zentai:', zentai)
print('hajime4:', hajime4)
print('ushiro4:', ushiro4)
print('mean_zentai:', mean_zentai)
print('mean_hajime4:', mean_hajime4)
print('mean_ushiro4:', mean_ushiro4)
```

```
print('kisu:', kisu)
print('gusu:', gusu)
```

実行結果を見てみましょう。

```
x = np.array([5.26, 4.78, 5.09, 4.57, 5.06, 5.07, 4.96, 5.10, 4.84, 4.97])
```

まずはじめに、x[1], x[2], …, x[10]に 5.26, 4.78, …, 4.97 を格納します。

```
# [1] 各種平均を計算
# 全体の平均
zentai = np.sum(x) / 10
# 最初の 4 つの平均
hajime4 = np.sum(x[:4]) / 4
# 最後の 4 つの平均
ushiro4 = np.sum(x[6:10]) / 4
```

sum(x[1:10])/10 について説明します。x[1:10]は x[1], x[2], …, x[10]の値の集合(ベクトル)を表します。これを sum()により合計を計算します。最後に sum(x[1:10])を 10 で割れば、x[1], x[2], …, x[10]の平均値となります。同様に初めの 5 個の値の平均値(hajime5)、後ろ 5 個の値の平均値(ushiro5)を計算しました。

```
> 全体の平均: 4.970000000000001
> 最初の 4 つの平均: 4.925
> 最後の 4 つの平均 : 4.967499999999999
```

計算結果を出力すると zentai=4.97, hajime5=4.925, ushiro5=4.9675 と計算できました。これらの結果は以下のように、mean()関数により計算することもできます。

```
> 全体の平均: 4.970000000000001
> 最初の 4 つの平均: 4.925
> 最後の 4 つの平均 : 4.967499999999999
```

奇数番目と偶数番目の平均値(kisu, gusu)を求めてみましょう。この場合、c()を使って、奇数番目と偶数番目の要素を定義します。奇数番目の要素の値の集合は x[c(1,3,5,7,9)]、偶数番目の要素の値の集合は x[c(2,4,6,8,10)]となります。

```
> 奇数番目の平均: 5.042
> 偶数番目の平均 : 4.898000000000001
```

それぞれの平均値は結果を見ると kisu=5.042, gusu=4.898 となりました。

ヒトの感覚で 5 秒を予測するという実験なので、それぞれの平均値(zentai=4.97, hajime5=4.925, ushiro5=4.9675、kisu=5.042, gusu=4.898)は、ほぼ 5 秒に近い値となっています。

標準偏差、不偏分散

不変分散とは、[測定データと平均値の差の 2 乗]を[サンプル数 N-1]で割った値です。式で書くと、

$$\mathrm{var(x)} = \frac{(x_1-\bar{x})^2+\cdots+(x_i-\bar{x})^2+\cdots+(x_N-\bar{x})^2}{9} = \frac{\sum_{i=1}^{N}(x_i-\bar{x})^2}{N-1}$$

となります。ここで、不偏分散（通常使われる）では分母は $N-1$ となります。
また、**標準偏差**とは、分散の平方根です。

$$\textbf{標準偏差} = \sqrt{\textbf{分散}}$$

$$\mathrm{sd(x)} = \sqrt{\frac{(x_1-\bar{x})^2+\cdots+(x_i-\bar{x})^2+\cdots+(x_N-\bar{x})^2}{9}} = \sqrt{\frac{\sum_{i=1}^{N}(x_i-\bar{x})^2}{N-1}}$$

1 回目から 10 回目の標準偏差（SDzentai）、はじめの 4 回の**標準偏差**（SDhajime）、後ろの 4 回の**標準偏差**（SDushiro）を計算するプログラムを作成してみましょう。

Pynarakita04

```
import numpy as np

# データの定義
x = np.array([5.26, 4.78, 5.09, 4.57, 5.06, 5.07, 4.96, 5.10, 4.84, 4.97])

# [1] 全体の平均と標準偏差を計算
# 全体の平均
Mzentai = np.mean(x)
# 全体のデータ数
nzentai = len(x)
# 平均からの偏差
difx = x - Mzentai
# 全体の標準偏差（不偏推定量）
SDzentai = np.sqrt(np.sum(difx**2) / (nzentai - 1))
print("全体の標準偏差:", SDzentai)

# [2] numpy の標準偏差関数を使って計算
# 全体の標準偏差（不偏推定量）
sd_zentai = np.std(x, ddof=1)
print("全体の標準偏差:", sd_zentai)

# 最初の 4 つの標準偏差
sd_hajime4 = np.std(x[:4], ddof=1)
print("最初の4つの標準偏差:", sd_hajime4)

# 最後の 4 つの標準偏差
sd_ushiro4 = np.std(x[6:10], ddof=1)
print("最後の4つの標準偏差:", sd_ushiro4)
```

[1]

Mzentai は x[1], x[2], …,x[10]の平均値です。

```
nzentai = len(x)
```

により、Mzentai の要素数を求めます。ここでは nzentai には 10 が格納されます。

```
difx = x - Mzentai
```

により、x[1], x[2], …,x[10]と Mzentai の差が計算されます。すなわち、difx[1], difx[2], …, difx[10]にはそれぞれ 0.29、-0.19、…、 0.00 となります。

```
np.sqrt(np.sum(difx**2) / (nzentai - 1))
```

「difx^2」で difx の要素それぞれについて 2 乗が計算されます。

sum(difx^2)では difx^2 の値の合計が計算できます。これを nzentai-1 で割り√を計算すれば SDzentai が得られます。

[2]では、sd()により x の標準偏差を計算しました。結果をみてみましょう。

```
全体の標準偏差: 0.196
最初の4つの標準偏差: 0.309
最後の4つの標準偏差: 0.106
```

式にもとづいて計算した標準偏差 SDzentai と sd(x)とした場合の標準偏差は一致しました(まあ当然！)そのうえで、最初の 4 回の標準偏差 0.309 に比べると、うしろ 4 回の実験の標準偏差が 0.106 と小さくなっていることがわかります。このように後半の方が、5 秒を当てる技能が向上していると考えられます。

二乗平均平方根誤差(Root Mean Squared Error)

二乗平均平方根誤差は、

$$\mathrm{RMSE}(x) = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{N}}$$

となります。ここで、$\hat{x}_i$は i 番目の測定値の真の値となります。「5 秒間」を予測する場合は、$\hat{x}_i = 5$ であるので

$$\mathrm{RMSE}(x) = \sqrt{\frac{\sum_{i=1}^{N}(x_i - 5)^2}{N}}$$

となります。RMSE について予測値全体を RMSEzentai、はじめの 4 個の予測値について RMSEhajime4、うしろ 4 個の予測値について RMSEushiro4 として Pynakita05 に実装しました。プログラムを解読してみてください。

Pynarakita05

```
import numpy as np

# データの定義
x = np.array([5.26, 4.78, 5.09, 4.57, 5.06, 5.07, 4.96, 5.10, 4.84, 4.97])
```

```
# 平均二乗誤差平方根(Root Mean Square Error: RMSE)を計算
# 全体の RMSE
RMSEzentai = np.sqrt(np.sum((x - 5.0)**2) / 10)
print("全体の RMSE:", RMSEzentai)

# 最初の 4 つの RMSE
RMSEhajime4 = np.sqrt(np.sum((x[:4] - 5.0)**2) / 4)
print("最初の 4 つの RMSE:", RMSEhajime4)

# 最後の 4 つの RMSE
RMSEushiro4 = np.sqrt(np.sum((x[6:10] - 5.0)**2) / 4)
print("最後の 4 つの RMSE:", RMSEushiro4)
```

実行結果をみると、

全体の RMSE: 0.18857359306117055

最初の 4 つの RMSE: 0.2779388421937457

最後の 4 つの RMSE: 0.09759610647971566

となり、RMSEushiro4 が、RMSEhajime4 に比べて、二乗平均平方根誤差が非常に小さくなっていることがわかります。このように単純な実験ですが、ヒトの感覚時間と実際の時間の間の誤差は訓練により向上したと考えられます。

箱ひげ図 (boxplot)

箱ひげ図により zentai、hajime4、ushiro4 の違いを見てみましょう。箱ひげ図の主な特長を以下に説明します。

・$Q_{1/4}$, $Q_{1/2}$, $Q_{3/4}$ は第1四分位、中央値(第2四分位)、第3四分位といいます。

・Low whiskey(Minimum)：$Q_{1/4} - 1.5 \cdot$ IQR であり、正規分布の場合 -2.698σ と対応。

ここで、IQR $= Q_{3/4} - Q_{1/4}$ です。

・Upper whiskey(Maximum)：$Q_{3/4} + 1.5 \cdot$ IQR であり、正規分布の場合 2.698σ と対応。

・外れ値:最小値より小さい値、あるいは最大値より大きな値であり、Low whiskey(Minimum)と Upper whiskey は、正規分布の場合 99.3%を占めるので、外れ値の出現確率は $p < 0.007$ 程度です。

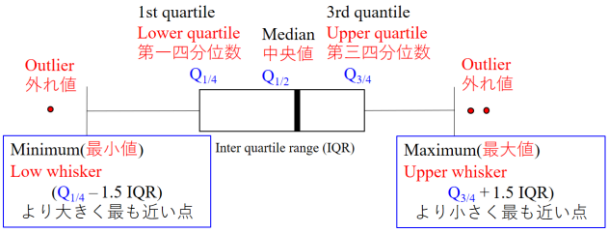
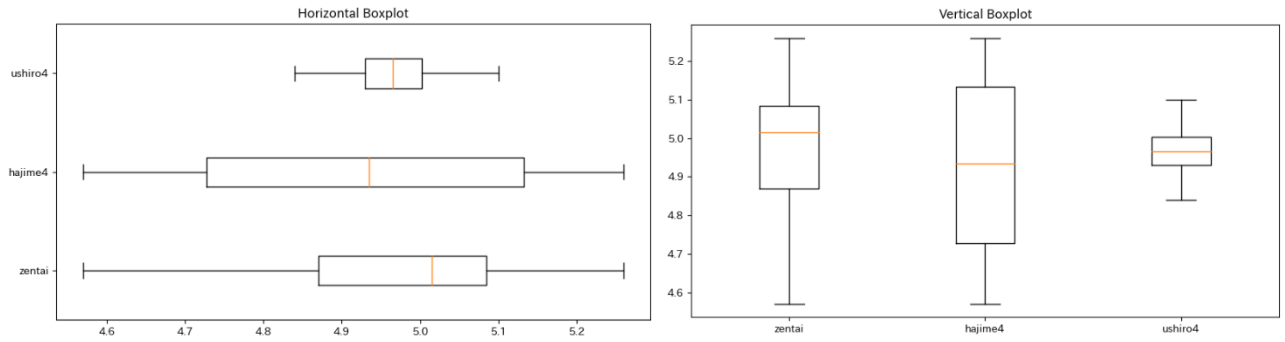


図 箱ひげ図(boxplot)の概要

Pynarakita06

```
import matplotlib.pyplot as plt
import numpy as np

# データの定義
x = [5.26, 4.78, 5.09, 4.57, 5.06, 5.07, 4.96, 5.10, 4.84, 4.97]

# 全体
zentai = x
# 最初の 4 つ
hajime4 = x[:4]
# 最後の 4 つ
ushiro4 = x[6:10]

# [1]
plt.figure(figsize=(10, 5))
plt.boxplot([zentai, hajime4, ushiro4], vert=False)
plt.yticks([1, 2, 3], ["zentai", "hajime4", "ushiro4"])
plt.title("Horizontal Boxplot")
plt.show()

# [2]
plt.figure(figsize=(10, 5))
plt.boxplot([zentai, hajime4, ushiro4], vert=True)
plt.xticks([1, 2, 3], ["zentai", "hajime4", "ushiro4"])
plt.title("Vertical Boxplot")
plt.show()

# [3]
median_zentai = np.median(zentai)
median_hajime4 = np.median(hajime4)
median_ushiro4 = np.median(ushiro4)

# 中央値を出力
print("全体の中央値:", median_zentai)
print("最初の 4 つの中央値:", median_hajime4)
print("最後の 4 つの中央値:", median_ushiro4)
```

[1][2]では、それぞれ横向きと縦向きに箱ひげ図を描画しました。また、[3]では中央値を求めました。



また、中央値については以下のようになります。

全体の中央値: 5.015
最初の 4 つの中央値: 4.9350000000000005
最後の4つの中央値: 4.965

この結果をみると、中央値で見ると微妙に ushiro4 が 5 秒より前に予測する傾向があり、hajime4 では 5 秒後に予測する傾向があるように見えます。平均値をみると平均値 (zentai=4.97, hajime5=4.92, ushiro5=4.97) でした。これを見ると hajime4 と ushiro4 の間には、平均値に差がないようです。でも、本当に hajime4 と ushiro4 の平均値に差はないのでしょうか。このことは t 検定による 2 群の平均値の差の検定で確認できます。

2 群の平均値の差の t 検定

2 群データのサンプル数 (自由度 $n-1$)、平均値と標準偏差により、t 分布により平均値の 95%信頼区間を求めることができます。いま $\boldsymbol{x}=\{x_1,x_2,x_3,x_4\}$ のデータの平均値の 95%信頼区間は

$$\mathrm{t}\Big(p=0.025,\text{自由度}=3\Big)=-\mathrm{Thr},\ \mathrm{t}\Big(p=0.975,\text{自由度}=3\Big)=\mathrm{Thr}$$

として真の平均値 μ の区間を求めると、

$$\bar{x}-\mathrm{Thr}\sqrt{\frac{U^2}{N}}<\mu<\bar{x}+\mathrm{Thr}\sqrt{\frac{U^2}{N}}$$

と計算できます (5 節 b 参照)。この式をもとに、全てのデータ、hajime4、ushiro4 について平均値の 95%信頼区間を求めてみましょう (Pynarakita07)。

Pynarakita07

```
import numpy as np
from scipy.stats import t, ttest_ind

# データの定義
x = np.array([5.26, 4.78, 5.09, 4.57, 5.06, 5.07, 4.96, 5.10, 4.84, 4.97])
hajime4 = x[:4]
ushiro4 = x[6:]

# [1]全データについて信頼区間の計算
ci_lower_all = np.mean(x) + np.std(x, ddof=1) * t.ppf(0.025, df=(10-1)) /
np.sqrt(10)
ci_upper_all = np.mean(x) + np.std(x, ddof=1) * t.ppf(0.975, df=(10-1)) /
np.sqrt(10)
print("全データについての信頼区間:", (ci_lower_all, ci_upper_all))

# [2]最初の 4 データについて信頼区間の計算
ci_lower_first4 = np.mean(hajime4) + np.std(hajime4, ddof=1) *
t.ppf(0.025, df=(4-1)) / np.sqrt(4)
ci_upper_first4 = np.mean(hajime4) + np.std(hajime4, ddof=1) *
t.ppf(0.975, df=(4-1)) / np.sqrt(4)
print("最初の4データについての信頼区間:", (ci_lower_first4, ci_upper_first4))


# [3]最後の 4 データについて信頼区間の計算
ci_lower_last4 = np.mean(ushiro4) + np.std(ushiro4, ddof=1) * t.ppf(0.025,
df=(4-1)) / np.sqrt(4)
ci_upper_last4 = np.mean(ushiro4) + np.std(ushiro4, ddof=1) * t.ppf(0.975,
df=(4-1)) / np.sqrt(4)
print("最後の4データについての信頼区間:", (ci_lower_last4, ci_upper_last4))

# [4]最初の 4 データと最後の 4 データの t 検定
t_test_result = ttest_ind(hajime4, ushiro4, equal_var=False)
print(t_test_result)

# [5]最初の 4 データと最後の 4 データの絶対誤差に関する t 検定(片側検定)
mseh4 = np.abs(hajime4 - 5)
mseu4 = np.abs(ushiro4 - 5)
t_test_mse_result = ttest_ind(mseh4, mseu4, alternative='greater',
equal_var=False)
print(t_test_mse_result)
```

結果をまとめると以下のようになり、いずれも実測値 5.0 を含む範囲となっています。

データセット	下限値	上限値
全てのデータ	4.830	5.110
hajime5	4.433	5.417
ushiro5	4.790	5.136

また、hajime4 と ushiro4 の平均値に有意な差があるか検定してみましょう。ここで、

帰無仮説 H_0 ： hajime4 と ushiro4 の平均値は等しい。

対立仮説 H_1 ： hajime4 と ushiro4 の平均値は等しくない。

となります。このことは、t.test(hajime4, ushiro4)により検定できます。結果をみてみましょう。

```
TtestResult(statistic=-0.26196313155315853, pvalue=0.805617459181417,
df=4.229520843375567)
```

帰無仮説 H_0 の p 値(0.8056)は十分大きいため、「hajime4 と ushiro4 の平均値は等しい」とみなすことができるということになります。通常 $p < 0.05$ のとき帰無仮説 H_0 は棄却されます。

[5]では、誤差(理論値はここでは 5 秒です)についての hajime4 と ushiro4 の検定を行っています。すなわち

$$\mathrm{mseh4} = \{|x_1 - 5|, |x_2 - 5|, |x_3 - 5|, |x_4 - 5|\}$$

$$\mathrm{mseu4} = \{|x_7 - 5|, |x_8 - 5|, |x_9 - 5|, |x_{10} - 5|\}$$

としました。この場合、対立仮説 $\mathbf{H_1}$ を mseh4 の平均値は mseu4 の平均値より大きいとしました。そこで ttest_ind()について alternative="greater"となっています。結果をみてみましょう。

```
TtestResult(statistic=2.1950505391470263, pvalue=0.046002000566035935,
df=4.069593189582943)
```

p 値は 0.046 であり、$p < 0.05$ となっているので、帰無仮説 H_0 は棄却できます。すなわち、対立仮説 H_1「mseh4 の平均値は mseu4 の平均値より大きい」となりました。つまり、この少ないデータであっても、ヒトによる時間感覚は、学習により予測精度が上がる(誤差は小さくなる)という仮説が成り立つことがわかりました。このように、統計検定を行うことで、p 値(確率値)により仮説の統計的有意性を説明することができます。このように、統計検定を使い、探究課題での仮説の統計的有意性を議論しましょう。

6.2 データの散らばり具合を表す指標

A チームと B チームでボウリングを行った。
A チームの 6 回のスコアは{183, 83, 33, 108, 133, 108}、
B チームの 6 回のスコアは{109, 122, 106, 100, 111, 108}
だった。

(1) それぞれのチームの平均値を求めてみよう。
(2) それぞれのチームの平均値について t 検定により有意な差があるか検討してみよう。
(3) それぞれのチームの標準偏差を求めてみよう。
(4) それぞれのチームのスコアの平均値との差の絶対値において t 検定により有意な差があるか検討してみよう。
(5) 二つのチームのスコアと、スコアと平均値の差、それぞれについて箱ひげ図を作成しよう。
(参考、Pynarakita09)

これらの結果から何が言えるか検討してみよう。また、標準偏差を社会活用する場面を考えてみよう。

Pynarakita09

```
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import ttest_ind

# データの定義
A = np.array([183, 83, 33, 108, 133, 108])
B = np.array([109, 122, 106, 100, 111, 108])

# [1] 各グループの平均を計算
mean_A = np.mean(A)
mean_B = np.mean(B)
print("各グループの平均:", mean_A, mean_B)


# [2] A と B のサンプルに対する t 検定
t_test_AB = ttest_ind(A, B)
print("A と B のサンプルに対する t 検定:", t_test_AB)

# [3] 各グループの標準偏差を計算
sd_A = np.std(A, ddof=1)
sd_B = np.std(B, ddof=1)
print("各グループの標準偏差:", sd_A, sd_B)

# [4] 各データ点の平均からの偏差の絶対値に対する t 検定
difA = np.abs(A - mean_A)
difB = np.abs(B - mean_B)
t_test_difAB = ttest_ind(difA, difB)
```

```
# ボックスプロットの表示
# [5] A と B のボックスプロット
plt.boxplot([A, B], labels=['A', 'B'])
plt.title('Boxplot of A and B')
plt.show()

# difA と difB のボックスプロット
plt.boxplot([difA, difB], labels=['difA', 'difB'])
plt.title('Boxplot of difA and difB')
plt.show()
```

6.3 2群の平均値の有意差検定：t 検定

ある昆虫のオスとメスの体重を 16 個体ずつ測定した（単位 mg）。
male(♂)={42,43,45,45,46,46,46,47,47,48,48,49,49,49,50,50}
female(♀)={42,42,43,43,43,44,44,44,45,45,45,45,46,47,48,49}
このときの2群の平均値の有意差を求めてみよう。

［参考、Pynarakita10、啓林館、理数探究基礎：未来に向かって、p.125］

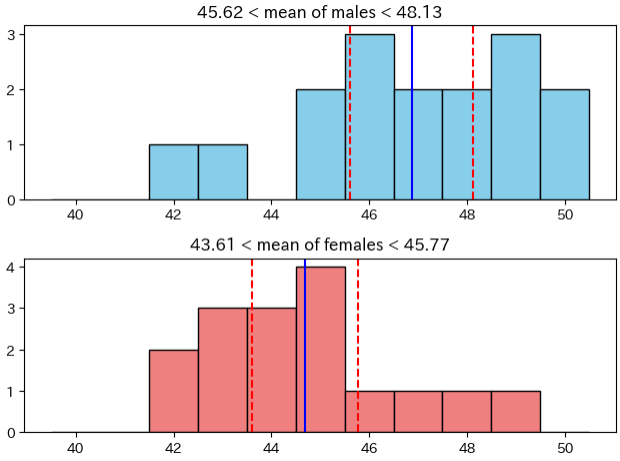
Pynarakita10

```
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import ttest_ind, t

# データの定義
male = np.array([42, 43, 45, 45, 46, 46, 46, 47, 47, 48, 48, 49, 49, 49,
50, 50])
female = np.array([42, 42, 43, 43, 43, 44, 44, 44, 45, 45, 45, 45, 46, 47,
48, 49])

# [1] 男性と女性のデータに対する t 検定を行います
t_test_result = ttest_ind(male, female, equal_var=False)
print("オスとメスのデータに対する t 検定:", t_test_result)

# [2] オスのデータに関する統計を計算します
mean_male = np.mean(male)
sd_male = np.std(male, ddof=1)
n_male = len(male)
t_value_male = t.ppf([0.025, 0.975], df=n_male-1)
ci_male = mean_male + sd_male / np.sqrt(n_male) * t_value_male
print("オスの平均:", mean_male)
print("オスの標準偏差:", sd_male)
print("オスの信頼区間:", ci_male)

# オスのデータのヒストグラムと信頼区間を描画します
plt.subplot(2, 1, 1)
plt.hist(male, bins=np.arange(39.5, 51.5, 1), color='skyblue',
edgecolor='black')
plt.title(f"{ci_male[0]:.2f} < mean of males < {ci_male[1]:.2f}")
plt.axvline(ci_male[0], color='red', linestyle='dashed')
plt.axvline(mean_male, color='blue')
plt.axvline(ci_male[1], color='red', linestyle='dashed')

# [3] 女性のデータに関する統計を計算します
mean_female = np.mean(female)
sd_female = np.std(female, ddof=1)
n_female = len(female)
t_value_female = t.ppf([0.025, 0.975], df=n_female-1)
```

```
ci_female = mean_female + sd_female / np.sqrt(n_female) * t_value_female
print("オスの平均:", mean_female)
print("オスの標準偏差:", sd_female)
print("オスの信頼区間:", ci_female)

# 女性のデータのヒストグラムと信頼区間を描画します
plt.subplot(2, 1, 2)
plt.hist(female, bins=np.arange(39.5, 51.5, 1), color='lightcoral',
edgecolor='black')
plt.title(f"{ci_female[0]:.2f} < mean of females < {ci_female[1]:.2f}")
plt.axvline(ci_female[0], color='red', linestyle='dashed')
plt.axvline(mean_female, color='blue')
plt.axvline(ci_female[1], color='red', linestyle='dashed')

# グラフを整えて表示します
plt.tight_layout()
plt.show()
```

t 分布を用いた区間推定

```
TtestResult(statistic=2.8124775791902996, pvalue=0.008684516635894108,
df=29.3078161359852)
```



コラム：`t.test()`関数を使って区間推定を行う

`X` は `16` 個のデータ{`42,43,45,45,46,46,46,47,47,48,48,49,49,49,50,50`}からなり、

平均値は$\bar{x} = 46.875$ `[mean(X)` で求められる`]`

$\sqrt{\frac{U^2}{N}}$ =`0.590727 [sqrt(var(X))` で求められる`]`

$t(p = 0.025, \text{自由度} = 16 - 1) = -2.13145$ `[qt(0.025,15)` で求められる`]`

$t(p = 0.975, \text{自由度} = 16 - 1) = 2.13145$ `[qt(0.975,15)` で求められる`]`

これらの値を使い

$$45.61589 < \mu < 48.13411$$

となります。

これは、`t.test(X,mu=mean(X))$conf.int` として求めることができる。その結果は、

```
45.6159 48.1341
```

となり、上記の計算とも一致します。

6.4 標準誤差と95%信頼区間

0.3%食塩水を与えて育てたトマトと、水を与えて栽培したトマトの果実400個ずつの糖度を測定した。このときのそれぞれの標準誤差、平均値の95%信頼区間の両側を求めてみよう。

	平均値	標準偏差
0.3%食塩水	10.27	1.8
水	6.08	0.8

[啓林館、理数探究基礎:未来に向かって p.43]

$$\bar{x} + \mathrm{t}\left(\mathrm{p} = 0.025, \text{自由度} = N - 1\right)\sqrt{\frac{U^2}{N}} < \mu < \bar{x} + \mathrm{t}\left(\mathrm{p} = 0.975, \text{自由度} = \mathrm{N} - 1\right)\sqrt{\frac{U^2}{N}}$$

ここで$U^2 = \frac{\sum_{i=1}^{N}(x_1 - \bar{x})^2}{N-1}$である。

食塩水の場合の標準誤差は

$$\sqrt{\frac{U^2}{N}} = \sqrt{\frac{1.8 \cdot 1.8}{400}} = 0.09$$

水の場合の標準誤差は

$$\sqrt{\frac{U^2}{N}} = \sqrt{\frac{0.8 \cdot 0.8}{400}} = 0.04$$

食塩水の場合の平均値の95%信頼区間は、

t(p=0.025,自由度=399)= -1.965927

t(p=0.975,自由度=399)= 1.965927

$$\bar{x} - 1.965927\sqrt{\frac{U^2}{N}} < \mu < \bar{x} + 1.965927\sqrt{\frac{U^2}{N}}$$

となるので
食塩水の場合の平均値の95%信頼区間は、

$$10.27 - 1.965927 \cdot 0.09 < \mu < 10.27 + 1.965927 \cdot 0.09$$

$$10.09 < \mu < 10.44$$

水の場合の平均値の95%信頼区間は、

$$6.08 - 1.965927 \cdot 0.04 < \mu < 6.08 + 1.965927 \cdot 0.04$$

$$6.00 < \mu < 6.15$$

となります。

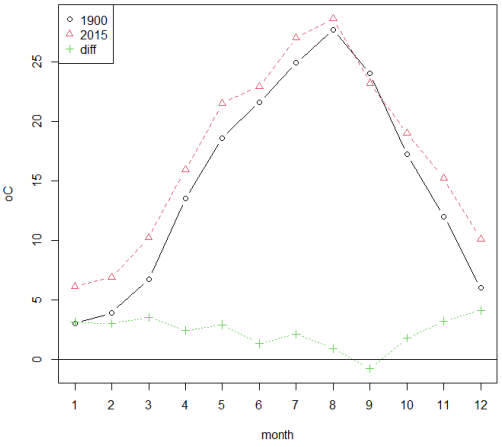
6.5 大阪市における月平均気温の比較を行おう

1900 年と 2015 年の大阪市の月ごとの平均気温を以下に示します。これらの気温を比較してなにが言えそうか検討してみましょう。

測定年	1 月	2 月	3 月	4 月	5 月	6 月	7 月	8 月	9 月	10 月	11 月	12 月
1900	3.0	3.9	6.7	13.5	18.6	21.6	24.9	27.7	24.0	17.2	12.0	6.0
2015	6.1	6.9	10.2	15.9	21.5	22.9	27.0	28.6	23.2	19.0	15.2	10.1

[啓林館、理数探究基礎:未来に向かって、p.40]

1900 年と 2015 年の月ごとの平均気温と、2015 年と 1900 年の月ごと平均気温の差をプロットしてみました。
1900 年(黒〇)、2015 年(赤△)、平均気温の差(＋緑)をみると、9 月以外は全て 2015 年の月平均気温の方が 1900 年より高くなっています。
それでは、対応のある t 検定を行ってみましょう。




```
> t.test(data1900,data2015,paired=TRUE,var.equal=FALSE,
  alternative="two.sided")
        Paired t-test
data:  data1900 and data2015
t = -5.9097, df = 11, p-value = 0.0001017
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -3.145162 -1.438171
sample estimates:
mean of the differences
              -2.291667
```

p-value = 0.0001017 と非常に小さく 1900 年と 2015 年では月ごとの平均値に統計的有意差があることがわかります。その温度差、2.29℃となっています。

この結果をもとに、これは大阪市だけの問題なのか、日本全体の問題なのか、などの解析をすすめようかなどと考えるのもアリでしょう。また、9 月はなぜ 2015 年にくらべて 1900 年の平均気温が高いのか？などの新たな課題を考えてみましょう。

コラム:6.4 の例では、2 群の平均値の差の検定(t 検定)では、
帰無仮説 H_0: $\mu_{2015} = \mu_{1900}$
として検定しました。対立仮説についてちょっと詳しく説明します。

$$\mu_{2015} = \mu_{1900}$$

ではないということは、

$$H_1 : \mu_{2015} < \mu_{1900} \quad (A)$$

$$H_1 : \mu_{2015} \neq \mu_{1900} \quad (B)$$

$H_1 : \mu_{2015} > \mu_{1900}$　(C)

の 3 つが考えられます。

(A), (B), (C)については、t.test()で、alternative="less","two.sided", "greater"とすることで、対応することができます。

```
t.test(data1900,data2015,paired=TRUE,var.equal=FALSE,alternative="less")
t.test(data1900,data2015,paired=TRUE,var.equal=FALSE,alternative="two.sided")
t.test(data1900,data2015,paired=TRUE,var.equal=FALSE,alternative="greater")
```

の結果を以下に示します。

```
> t.test(data1900,data2015,paired=TRUE,var.equal=FALSE,alternative="less")
        Paired t-test
data:  data1900 and data2015
t = -5.9097, df = 11, p-value = 5.084e-05
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
     -Inf -1.59526
sample estimates:
mean of the differences
              -2.291667
> t.test(data1900,data2015,paired=TRUE,var.equal=FALSE,alternative="two.sided")
        Paired t-test
data:  data1900 and data2015
t = -5.9097, df = 11, p-value = 0.0001017
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -3.145162 -1.438171
sample estimates:
mean of the differences
              -2.291667
> t.test(data1900,data2015,paired=TRUE,var.equal=FALSE,alternative="greater")
        Paired t-test
data:  data1900 and data2015
t = -5.9097, df = 11, p-value = 0.9999
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
 -2.988073       Inf
sample estimates:
mean of the differences
              -2.291667
```

$\bar{x}_{1900}(= 14.93) < \bar{x}_{2015}(= 17.21)$ であることを考慮して検定結果をみましょう。

(A)では、対立仮説は $H_1 : \mu_{2015} > \mu_{1900}$ であるので、
区間推定としては、$-\infty$ から -1.59526 が $\mu_{1900} - \mu_{2015}$ となります。
この場合、p 値は(A) 、(B) 、(C)の中で最小の p 値となります。

(B)では、$H_1 : \mu_{2015} > \mu_{1900}$ と $\mu_{2015} < \mu_{1900}$ の両方とも検定しています。
区間を推定すると $-3.145162 < \mu_{1900} - \mu_{2015} < -1.438171$ となります。
(A)と(B)の推定区間の閾値は全て－なので 0 を含みません。
それで $\mu_{1900} - \mu_{2015} = 0$ ではないということを示しています。

(C)では、$H_1 : \mu_{2015} < \mu_{1900}$ です。解析したデータではこれは成り立ちません。

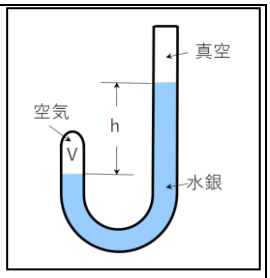
(c) の対立仮説 H_1 としての区間推定は、$-2.988073<\mu_{1900}-\mu_{2015}<\infty$ となり 0 を含んでおり、$p>0.05$ となっています。(A) と (C) を**片側検定**、(B) を**両側検定**といいます。3 種の帰無仮説のどれを使うかはユーザーに任されています。状況に応じて、適切に使いましょう。

	H_1	H_0 の p 値	区間推定	t.test() の alternative の設定
(A)	$\mu_{2015}>\mu_{1900}$	0.00005084	-Inf< $\mu_{1900}-\mu_{2015}$ < -1.59526	less
(B)	$\mu_{2015}\neq\mu_{1900}$	0.0001017	-3.145162<$\mu_{1900}-\mu_{2015}$ < -1.438171	both.side
(C)	$\mu_{2015}<\mu_{1900}$	0.9999	-2.988073<$\mu_{1900}-\mu_{2015}$ <Inf	greater

6.6 回帰モデル:データから法則性をみいだそう

気体の体積と圧力の関係

かつて 17 世紀にイギリスの科学者ボイルは、気体の体積と圧力の関係を調べる実験を行った。J 字状の管の先端に閉じ込めた空気の体積 V と、管に入れた水銀の量の関係を調べた。温度は一定である。液面の高さの差 h から閉じ込められた空気にかかる圧力 p を求めた (表)。この表をもとに p と V の関係を求めてみよう。



体積 V (cm³)	48.0	40.0	32.0	24.0	20.0	16.0	12.0
圧力 p (x10³hPa)	1.01	1.23	1.54	2.04	2.46	3.04	4.09

[啓林館、理数探究基礎:未来に向かって p.44]

法則性を見出すときには、とりあえず対数を取りプロットをするのもありです。底を 10 として二つの変数 V と p の対数をとり、直線関係が得られたとします。

$$\log_{10}V=a+b\log_{10}p$$

この式は、

$$\log_{10}V+\log_{10}p^{-b}=a$$

となり、さらに変形すると

$$\log_{10}(Vp^{-b})=a$$

$$Vp^{-b}=10^{a}$$

と変形できるため、a と b が得られれば V と p の関係を式で表すことができます。

ちょっと練習

(a) いま、x と y が比例関係にあるとき

$$y=x$$

対数をとると、傾き 1、切片 (定数) 0 の直線が得られます。

$$\log_{10}y=\log_{10}x$$

$$\log_{10}\frac{y}{x}=0$$

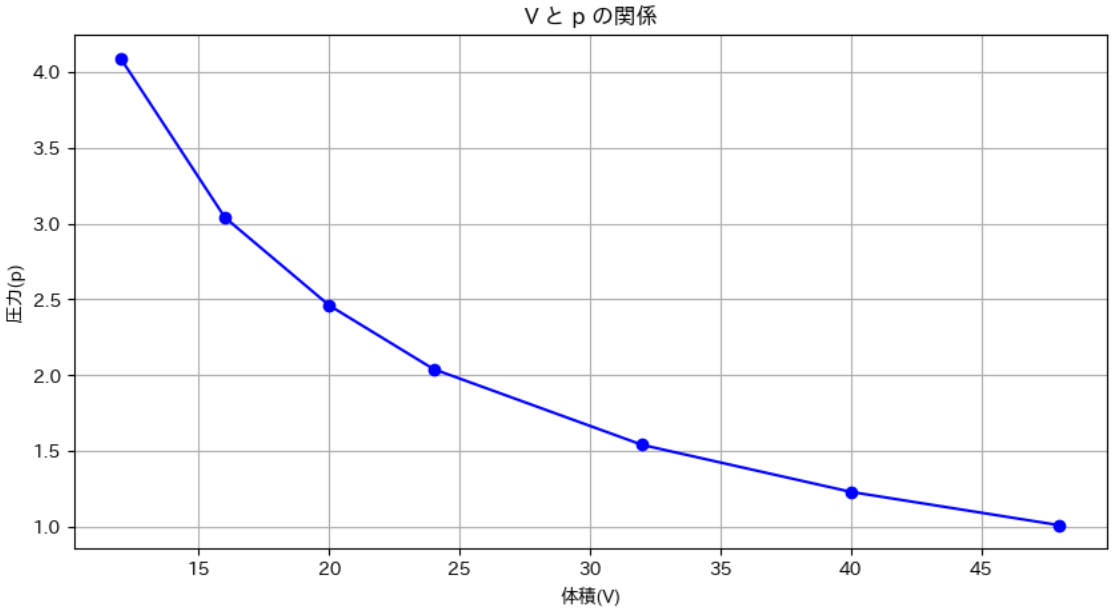
$$\frac{y}{x}=10^{0}=1$$

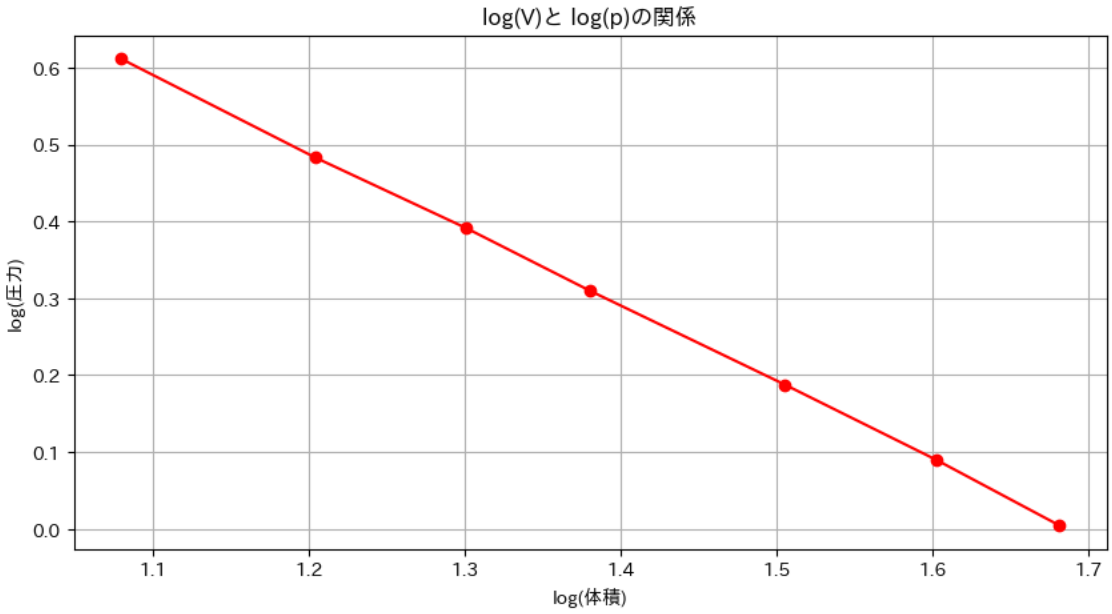
となります。
(b) $y = x^m$のとき
$\log_{10} y = m \log_{10} x$ となるため、y と x の傾きが m 、切片(定数)は 0 となります。よって、傾きから m を求めることができます。

実際に、[1] V と p を、x 軸と y 軸に設定しプロットしてみましょう。また、[2] $\log_{10} V$と $\log_{10} p$ を、x 軸と y 軸に設定しプロットしてみましょう。

Pynarakita07

```
Import japanize_matplotlib
import matplotlib.pyplot as plt
import numpy as np

# データの定義
V = np.array([48.0, 40.0, 32.0, 24.0, 20.0, 16.0, 12.0])
p = np.array([1.01, 1.23, 1.54, 2.04, 2.46, 3.04, 4.09])

# [1] V と p のプロットを作成し、点と点を線でつなぎます
plt.figure(figsize=(10, 5))
plt.plot(V, p, marker='o', color='blue')
plt.title('V と p の関係')
plt.xlabel('体積(V)')
plt.ylabel('圧力(p)')
plt.grid(True)
plt.show()

# [2] log(p)と log(V)のプロットを作成します
logV = np.log(V)
logp = np.log(p)
plt.figure(figsize=(10, 5))
plt.plot(logV, logp, marker='o', color='red')
plt.title('log10(V)と log10(p)の関係')
plt.xlabel('log10(体積)')
plt.ylabel('log10(圧力)')
plt.grid(True)
plt.show()
```





上側の図が[1] V と p の関係、右側の図が[2] $\log_{10}V$ と $\log_{10}p$ の関係となります。この図をみていただくと、$\log_{10}V$ と $\log_{10}p$ の間には、

$$\log_{10}V = a - b\log_{10}p$$

の関係があることがわかります。ここで、a と b をどのように求めるのでしょうか。ここでよく使われるのが最小二乗法です。

最小二乗法

いま、y と x の測定値があるとします。ここで、y と x の間には、

$$y = a + bx \qquad (1)$$

という関係があるとします。N 個の測定値 $y = \{y_1, y_2, \cdots, y_i, \cdots, y_N\}$ と $x = \{x_1, x_2, \cdots, x_i, \cdots, x_N\}$ を (1) に代入します。

$$y_1 = a + bx_1 + e_1$$
$$\cdots$$
$$y_i = a + bx_i + e_i$$
$$\cdots$$
$$y_N = a + bx_N + e_N$$

ここで、e_iは残差です。測定値 y_i と x_i がぴったり (1) の式にあうことはほぼありません。そこで、残差を入れて式を表します。では、この式を残差 e_i により式を変形してみましょう。

$$e_i = y_i - (a + bx_i)$$

$i = 1, 2, 3, \cdots, N$

そこで、全体の 2 乗誤差を定義します。

$$E(a, b) = {e_1}^2 + {e_2}^2 + \cdots + {e_N}^2 = \sum_{i=1}^{N} {e_i}^2$$

この式の $E(a, b)$ を最小とする a, b を求めます。

コラム：$E(a, b)$ を最小にする係数 a, b を手計算で求めるには

$$E(a, b) = {e_1}^2 + {e_2}^2 + \cdots + {e_N}^2 = \sum_{i=1}^{N} {e_i}^2$$

この式に、$e_i = y_i - (a + bx_i)$ を代入してみましょう。

$$E(a, b) = \{y_1 - (a + bx_1)\}^2 + \cdots + \{y_i - (a + bx_i)\}^2 + \cdots + \{y_N - (a + bx_N)\}^2$$
$$= \sum_{i=1}^{N} \{y_i - (a + bx_i)\}^2$$

$E(a, b)$ を最小にする係数 a, b を求めてみましょう。

最後の $\sum_{i=1}^{N} \{y_i - (a + bx_i)\}^2$ の項を見てみましょう。

$$\mathrm{E(a, b)} = \sum_{i=1}^{N} \{y_i - (\mathrm{a} + \mathrm{b}x_i)\}^2 = \sum_{i=1}^{N} \{y_i - \mathrm{a} - \mathrm{b}x_i\}^2$$

$$= \sum_{i=1}^{N} \{{y_i}^2 + a^2 + (\mathrm{b}x_i)^2 - 2ay_i - 2bx_i y_i + 2abx_i\}$$

$$= \sum_{i=1}^{N} y^2 + \sum_{i=1}^{N} a^2 + b^2 \sum_{i=1}^{N} {x_i}^2 - 2a \sum_{i=1}^{N} y_i - 2b \sum_{i=1}^{N} x_i y_i + 2ab \sum_{i=1}^{N} x_i$$

ここで、$\sum_{i=1}^{N} y^2$、$\sum_{i=1}^{N} y_i$、$\sum_{i=1}^{N} {x_i}^2$、$\sum_{i=1}^{N} x_i$、$\sum_{i=1}^{N} x_i y_i$ は実験値から計算できるので定数とおくことができます。

$$Y_2 = \sum_{i=1}^{N} y^2 \text{、} Y_1 = \sum_{i=1}^{N} y_i \text{、} X_2 = \sum_{i=1}^{N} {x_i}^2 \text{、} X_1 = \sum_{i=1}^{N} x_i \text{、} Z = \sum_{i=1}^{N} x_i y_i$$

また、$\sum_{i=1}^{N} a^2 = a^2 + a^2 + \cdots + a^2 = Na^2$ となります。

$$E(a, b) = Y_2 + Na^2 + b^2 X_2 - 2aY_1 - 2bZ_1 + 2abX_1$$

この式をみると、$E(a,b)$ は変数 a と b についての 2 次式となります。a^2 と b^2 の係数は N と $X_1 = \sum_{i=1}^{N} x_i$ はともに正なので最小値をもちます。これを簡単に求めるには、
$E(a,b)$ を片方の変数 b を固定して、a で微分した式を作り 0 とおきます。これを偏微分といい、$\frac{\partial E(a,b)}{\partial a}$
であらわします。

$$\frac{\partial E(a,b)}{\partial a} = 2Na - 2Y_1 + 2X_1 b = 0$$

同様に、変数 a を固定して、b で微分した式を作り 0 とおきます。

$$\frac{\partial E(a,b)}{\partial b} = 2X_2 b - 2Z_1 + 2X_1 a = 0$$

これら二つの式をもとに a，b を算出します。
すると、

$$a = \frac{X_2 Y_1 - X_1 Z}{N X_2 - {X_1}^2} = \frac{\sum_{i=1}^{N} {x_i}^2 \sum_{i=1}^{N} y_i - \sum_{i=1}^{N} x_i \sum_{i=1}^{N} x_i y_i}{N \sum_{i=1}^{N} {x_i}^2 - \left(\sum_{i=1}^{N} x_i\right)^2}$$

$$b = \frac{X_1 Y_1 - N Z_1}{N X_2 - {X_1}^2} = \frac{\sum_{i=1}^{N} x_i \sum_{i=1}^{N} y_i - N \sum_{i=1}^{N} x_i y_i}{N \sum_{i=1}^{N} {x_i}^2 - \left(\sum_{i=1}^{N} x_i\right)^2}$$

と求まります。

Python スクリプトにより a, b の係数を求めてみましょう。

$$\log_{10} V = a + b \log_{10} p$$

のように、$y = a + bx$ として a, b を求め、このモデルに従って、任意の x から y を予測するモデルのこと線形回帰モデル(linear regression model)といいます。Python プログラムでは、[1]のように関数 lm()に x と y とをデータセット指定すると簡単に求めることができます。確認のために[2]コラムで導いた方法でも求めてみよう。

Pynarakita08

```
import numpy as np
import pandas as pd
import statsmodels.api as sm
import statsmodels.formula.api as smf

# データの定義
V = np.array([48.0, 40.0, 32.0, 24.0, 20.0, 16.0, 12.0])
p = np.array([1.01, 1.23, 1.54, 2.04, 2.46, 3.04, 4.09])
logV = np.log10(V)
logp = np.log10(p)

# [1] 回帰モデルの作成
# データフレームの作成
dataYX = pd.DataFrame({'logV': logV, 'logp': logp})
print(dataYX)

# 回帰モデルのフィッティング
reg = smf.ols('logV ~ logp', data=dataYX).fit()
# モデルのサマリー
```

```
reg_summary = reg.summary()
print("モデルのサマリー")
print(reg_summary)
# 95%信頼区間の計算
reg_conf = reg.conf_int(alpha=0.05)
print("モデルの信頼区間")
print(reg_conf)
# 係数
a1 = reg.params['Intercept']
b1 = reg.params['logp']

# [2] 微分から導出
# サンプルサイズ
N = len(V)
# 係数 a2(切片)の計算
a2 = (np.sum(logV**2) * np.sum(logp) - np.sum(logV) * np.sum(logp * logV))
/ (N * np.sum(logV**2) - np.sum(logV)**2)
# 係数 b2(傾き)の計算
b2 = (N * np.sum(logV * logp) - np.sum(logV) * np.sum(logp)) / (N *
np.sum(logV**2) - np.sum(logV)**2)

# 結果の出力
print(a1, b1, 10**a1)
print(a2, b2, 10**a2)
```

logp には $\log_{10} p$ ， logV には $\log_{10} V$ の値が格納されています。データを data.frame 型にしないと lm()関数で回帰モデルを作ることができないので、とりあえず data.frame(logV, logp)として dataYX に代入します。dataYX には以下のように、7 サンプル 1-7 について logV と logp の値が格納されています。

```
> print(dataYX)
      logV      logp
0  1.681241  0.004321
1  1.602060  0.089905
2  1.505150  0.187521
3  1.380211  0.309630
4  1.301030  0.390935
5  1.204120  0.482874
6  1.079181  0.611723
```

logV を y（目的変数）、logp を x（説明変数）として、回帰モデルを作るスクリプトは、

```
smf.ols('logV ~ logp', data=dataYX).fit()
```

となります。`smf.ols('logV ~ logp', data=dataYX)`関数の中を見てみましょう。「~」左側が目的変数、右側が説明変数になります。ここでは、$\log_{10} V = a + b \log_{10} p$ という回帰モデルを作ります。得られた回帰モデルは `reg` に格納されます。

```
reg.summary()
```

により、回帰モデルを得ることができます。また

```
reg.conf_int(alpha=0.05)
```

により、係数 a, b の 95%信頼区間が得られます。

結果を見てみましょう。`Residuals` はそれぞれのサンプルの残差を表しています。また、`Coefficients` から、

$$\log_{10} V = 1.6895 - 0.9982 \log_{10} p$$

というモデルができたことがわかります。ここで、係数の右側にある $P > |t|$ は係数の統計有意水準を表しています。今回は $P > |t|$ が `0` なので、とってもいいモデルができたことを表しています。

`Prob (F-statistic)`とは回帰モデル全体における統計有意性を示す指標です。これも `1.00e-10` と非常に小さい `p` 値なので $\log_{10} p$ と $\log_{10} V$ には、線形回帰モデルとして関連づいていることがわかります。

```
> print(reg_summary)
OLS Regression Results
==============================================================================
==
Dep. Variable:                    logV   R-squared:
1.000
Model:                            OLS   Adj. R-squared:
1.000
Method:                 Least Squares   F-statistic:
3.241e+04
Date:                Mon, 16 Sep 2024   Prob (F-statistic):           1.00e-
10
Time:                        10:13:28   Log-Likelihood:
32.026
No. Observations:                   7   AIC:                             -
60.05
Df Residuals:                       5   BIC:                             -
60.16
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
==
```

```
                  coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
--
Intercept      1.6895      0.002    850.113      0.000       1.684
1.695
logp          -0.9982      0.006   -180.038      0.000      -1.012      -
0.984
==============================================================================
==
Omnibus:                          nan   Durbin-Watson:
2.125
Prob(Omnibus):                    nan   Jarque-Bera (JB):
0.763
Skew:                          -0.491   Prob(JB):
0.683
Kurtosis:                       1.714   Cond. No.
5.43
==============================================================================
==
```

続いて、`confint(reg,level=0.95)`により係数 a, b の区間推定を行ってみましょう。95%信頼区間において、

切片 a は、$1.684345 < a < 1.694563$

係数 b は $-1.012459 < b < -0.983954$

となりました。

注目すべきは、係数 b は -1 とみなすこともできるという点です。

```
> print(reg_conf)
                 0         1
Intercept  1.684345  1.694563
logp      -1.012459 -0.983954
```

なお`[1]`関数`lm()`にて x と y から求めた係数 a_1（切片）、`b1`（傾き）と、`[2]`コラムで導いた方法でも求めた係数を a_2（切片）、b_2（傾き）とすると

```
a1: 1.6895
b1: -0.9982
a2: 1.6923
b2: -1.002
10^a1: 48.916
```

```
10^a2: 49.235
```
となる。すなわち、$a_1 = a_2,\ b_1 = b_2$ となることが確認できた。

それでは、

$$\log_{10} V = 1.6895 - 0.9982 \log_{10} p$$

の式から p と V の関係を導いてみましょう。

$$\log_{10} p = 1.6925 - 1.0018 \log_{10} V$$

$$\log_{10} p + \log_{10} V^{1.0018} = 1.6925$$

$$\log_{10} pV^{1.0018} = 1.6925$$

$$pV^{1.0018} = 10^{1.6925} = 49.26$$

いま、係数 1.0018 は 95%信頼区間で 1 とみなせるので、おおよそ、この実験では、

$$PV = 49.26 (= \text{一定})$$

という結果が得られました。ちなみに、$PV =$ [一定] という気体の状態方程式は化学で扱う式です。

ピアソンの相関係数

N 個の測定値 $y = \{y_1,\ y_2, \cdots, y_i, \cdots, y_N\}$ と $x = \{x_1, x_2, \cdots, x_i, \cdots, x_N\}$ について、

$$r(x,y) = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2 \sum_{i=1}^{N}(y_i - \bar{y})^2}}$$

を変数 x と変数 y のピアソン相関といいます。ここで、$\bar{x}$ と $\bar{y}$ は、それぞれ x と y の平均値です。

ピアソン相関係数は、二つの測定値 x , y について平均値を引いた値により定義されています。いま

$$X_i = x_i - \bar{x}$$

$$Y_i = y_i - \bar{y}$$

と定義すれば、

$$r(X,Y) = \frac{\sum_{i=1}^{N} X_i Y_i}{\sqrt{\sum_{i=1}^{N} {X_i}^2 \sum_{i=1}^{N} {Y_i}^2}}$$

となります。この式は、二つのベクトル X と Y の内積とみなすことができて、

$$\cos(X,Y)=\frac{XY}{|X||Y|}$$

と同じであることが類推できるかと思います。$\cos\theta$ は -1 から 1 の範囲にあるので、相関係数も -1 と 1 の範囲にあります。この $r(X,Y)=1$ のとき右上がりの直線を表し、$r(X,Y)=-1$ のとき右下がりの直線を表します。そして、二つの変数に関係がないときは 0 となります。つまり、1 に近づくほど正の相関が高く、-1 に近づくと負の相関が高いということになります。

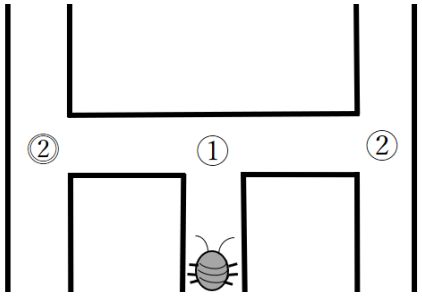
相関係数は `cor()` で簡単に求めることができます。`V` と `p`, `logV` と `logp` の相関係数を求めてみると、`logV` と `logp` の相関係数は`-0.9999229` となり、`V` と `p` の相関係数`(-0.9259517)`に比べて非常に強い負の相関があることがわかります。

```
> cor(V,p)
[1] -0.9259517
> cor(logV,logp)
[1] -0.9999229
```

このように回帰分析、相関係数などを用いると変数間の関係を経験的に求めることができます。ここで経験的といっているのは、いまあるデータからの数理モデルが導入できたというだけで、理論モデルではありません。ただ、経験的でない意味で良好な数理モデルができたとすると、これをもとに本来の理論を探究するきっかけとなるでしょう。 `Good luck!`

6.7 実験モデルの統計的有意性を検討する：χ２乗検定

「ダンゴムシは、図のような迷路を歩かせると最初に①で右に曲がると次は②で左に曲がる。また、①で左に曲がると②では右に曲がるという交替性転向反応を示す。」という。実験をダンゴムシで 100 回行った。63 回は 1 回目と違う向き、37 回は 1 回目と同じ向きに曲がった。この場合、交替性転向反応はあるといっていいだろうか。



［啓林館、理数探究基礎：未来に向かって p.104, p.124］

Python スクリプトでプログラム実装をしてみよう。

統計検定は、まず帰無仮説 H_0（二つの事象が同じであると仮定）をたてます。この場合、

H_0 ： 実験頻度が理論確率に従う。

とします。この H_0 が起こる確率 p を、確率分布をもとに計算します。いま、$p < 0.05$ のとき、H_0 が起こる確率は十分低いと判断し、対立仮説、

H_1 ：実験頻度が理論確率に従わない。

と判断します。では実際に Pynarakita02 によりダンゴムシの実験を判定してみましょう。

Pynarakita02

```
import numpy as np
from scipy.stats import chisquare

# 観測値と期待値の割合を定義
Obs = np.array([63, 37])
pv = np.array([0.5, 0.5])

# カイ二乗検定を実行
chisq_result = chisquare(Obs, f_exp=Obs.sum() * pv)

# 結果を出力
chisq_result
```

ここで、chisquare(x, p) では、x と p をそれぞれ実験頻度、ならびに理論確率とします。いま、Obs には、二つの事象、交替性転向反応が起こったサンプル数 63 と、起こらなかったサンプル数 37 からなるベクトルを定義します。次にランダムに曲がった場合の交替性転向反応が起こる確率と起こらない確率は、それぞれ 0.5 である。これを pv として定義する。

Obs と pv を chisquare(x, p) に代入すれば、帰無仮説 H_0（実験頻度が理論確率に従う）の確率を求めることができる。

実際に Pynarakita02 を実行すると以下の結果が得られる。

```
Power_divergenceResult(statistic=6.76, pvalue=0.009322)
```

実行結果を見ると p-value=0.009322 が、帰無仮説 H_0 の起こる確率である。よって、H_0 は棄却され、H_1（実験頻度が理論確率に従わない）となる。すなわち、ダンゴムシには、統計的有意水準により交替性転向反応が起こると結論づけられる。なお文献[1]にオカダンゴムシ(Armadillidium vulgare)での交替性転向反応の研究がなされているので、統計検定をしてみるといいと思う。

では、なぜ、交替性転向反応を起こす動物がいるのだろうか？ いまのところ二つの仮説で説明されています(文献[2])。

仮説 1：接触性仮説、オカダンゴムシは壁に触れながら歩き、曲がり角では触れていた方向に斜めに移動する。そのため、前とは逆側の体が壁に接触し、その壁との接触を保ったまま前進する。

仮説 2：角を曲がる際、カーブの外側の脚の作業量が内側の脚の作業量より大きい。この作業量差を平均化するため，交互に転向する。

さらに、文献[2]では、被験体を強制的に転向させる強制転向点(F)，被験体に左右の迷路を選択させる選択点(C)を設けた T 字迷路における FC 間の距離 32～36 cm の間に交替性転向の限界があることを示している。

文献

[1] 渡辺宗孝、岩田清二、ダンゴムシにおける交替性転向反応、Annual Animal Psychol., 6, 75-81 (1956)

[2] 草野ゆうか，新妻裕翼(顧問：小平裕子，福島県立磐城高等学校) オカダンゴムシの交替性転向の仕組みを探る、化学と生物、53, 130-132, (2015)

高校生からのコメント（思考実験）

セミの成虫の寿命は、一般には一週間といわれる。これを統計検定により明らかにする方法を考えてみよう。ここでポイントになるのは、成虫になったところから、死ぬまでの期間をいかに高精度で測定するかである。そのために、成虫になった瞬間のセミに行動に支障をきたさないマイクロチップをつけ、GPS により行動を観察することを考えた。単に GPS で追跡するのだと、あまりに広範になるので、セミが生きていくために支障のない、閉鎖系で、追跡できれば、「セミ成虫の寿命が一週間である」説を検証できる。仮に 100 匹のセミ成虫を追跡して、95 匹以上が一週間以内に死亡すれば、この仮説は成り立つとして、統計検定してみるというのを思いついた。

6.8 水生物調査から地域の河川の水質について考える

大阪湾にそそぐ A-J の 10 地点で水生生物調査と、水質調査を行った。水生生物の出現特性と水質調査の地点の類似性を検討してみよう。

[啓林館、理数探究基礎:未来に向かって p.82]

▲ 図2 調査地

▼ 表1 槇尾川の水生生物調査結果（2018年8月 1地点につき5か所、各5分間に採取した個体数の合計*2）

地点	A	B	C	D	E	F	G	H	I	J
サワガニ	4	3		2						
カゲロウ	6	9	7	4	4	6	5	2	3	
ヘビトンボ		3			15	5				
トビケラ	3	5		2	2	3				
カワニナ	4	3		5						
ミズムシ										
ヒル			1		3		2	1		
サカマキガイ			2		4				17	
イトミミズ							3		7	14
ユスリカ							7	6		13

▼ 表2 槇尾川の水質調査の結果（ND は検出できなかったことを表す）

	A	B	C	D	E	F	G	H	I	J
COD（ppm）	5	5	10	10	20	10	20	20	10	50
PO_4^{3-}（ppm）	0.02	0.02	0.02	0.02	0.05	0.02	0.1	0.1	0.1	0.1
NH_4^+（ppm）	ND	0.2	0.2	ND	0.2	0.2	0.5	0.5	0.5	0.5

生物種については、ミズムシを除けば、9 種類の生物の出現数が測定されています。一方、水質については 3 種の化学分析が行われた結果が得られています。

まずデータ行列をつくりましょう。

```
> dataSet
  sawa kage hebi tobi kawa hiru saka ito yusu COD  PO4 NH4
A    4    6    0    3    4    0    0   0     0   5 0.02  NA
B    3    9    3    5    3    0    0   0     0   5 0.02 0.2
C    0    7    0    0    0    1    2   0     0  10 0.02 0.2
D    2    4    0    2    5    0    0   0     0  10 0.02  NA
E    0    4   15    2    0    3    4   0     0  20 0.05 0.2
F    0    6    5    3    0    0    0   0     0  10 0.02 0.2
G    0    5    0    0    0    2    0   3     7  20 0.10 0.5
H    0    2    0    0    0    1    0   0     6  20 0.10 0.5
I    0    3    0    0    0    0   17   7     0  10 0.10 0.5
J    0    0    0    0    0    0    0  14    13  50 0.10 0.5
```

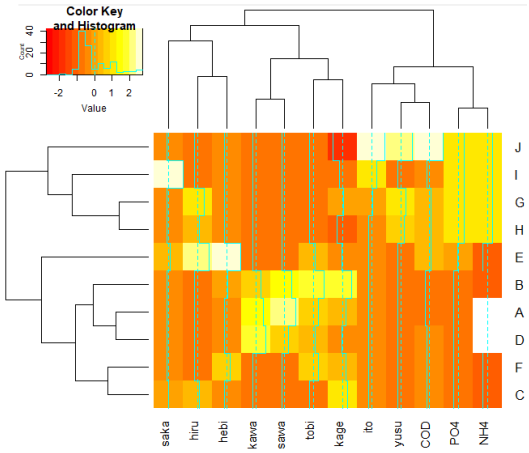
ここで、生物名を以下のように略した。sawa（サワガニ）， kage（カゲロウ）， hebi（ヘビトンボ）， tob（トビケラ）, kawa（カワニナ）， hiru（ヒル）， saka（サカマキガイ）， ito（イトミミズ）， yusu（ユスリカ）。また 3 種の分析についても以下のように略した。COD（COD [ppm]）、PO4（PO_4^{3-}[ppm]）、NH4（NH_4^+[ppm]）。

このように、それぞれの測定の単位が異なる場合は、変数ごとに、スケーリングし、ダイナミックレンジをそろえて解析する場合が多い。

$$x'_{ij} = \frac{x_{ij} - \bar{x}_j}{\mathrm{sd}(x_j)}$$

ここで、j 番目の変数の平均値を $\bar{x}_j$、標準偏差を $\mathrm{sd}(x_j)$ と表しています。こうすると、どの変数についても標準偏差は 1、平均値は 0 となります。このようにスケーリングした行列 DS を変数間とサンプル間の類似性をクラスター分析法で検討してみましょう。

クラスター分析法とは、変数についてみると、最も類似度が高い変数からつないでいき最終的にひとつのグループにまとめる方法です。同様に、サンプルについても同様の解析を進めることができます。さらに、両方のクラスター分析の結果と、行列の値のヒートマップをみれば、データの大小関係から変数、サンプルの関係を鳥瞰的に理解することができます。



結果をみてみましょう。サンプルを収集した地点をみると、(H, G, I, J)、(A, B, C, D, F) 、(E) において、出現する生物種と 3 種の分析データを含めて、類似の状況にあることを示しています。これは明らかに、上流と下流で川の状態が大きく異なることを示しています。

一方、生物種と分析法については以下の二つのグループ A と B に識別することができます。

A 群、[saka (サカマキガイ)、hiru (ヒル)、hebi (ヘビトンボ)]

B 群、[kawa (カワニナ)、sawa (サワガニ)、tob (トビケラ)、kage (カゲロウ)]、

C 群、[ito (イトミミズ)、yusu (ユスリカ)、 COD (COD [ppm])、PO4 (PO_4^{3-}[ppm])、NH4 (NH_4^+[ppm])]

[1] 4 つの地点 (H, G, I ,J) では C 群の生物種の出現頻度が高く、COD (COD [ppm])、PO4 (PO_4^{3-}[ppm])、NH4 (NH_4^+[ppm]) の値が高くなる。

[2] E 地点では A 群の生物種の出現頻度が高い。

[3] A-F 地点では、B 群の生物種の出現頻度が高い。

文献[1]で示されている指標種に、本研究で出現した生物種にグループ名とともに下線をつけてみました。

表 日本水環境学会 (2006) による水質と指標生物の関係

水質	指標生物
きれいな水	アミカ・ウズムシ・カワゲラ・サワガニ (B)・ナガレトビケラ (B)・ヒラタカゲロウ (B)・ブユ・ヘビトンボ・ヤマトビケラ (B)
少し汚い水	イシマキガイ・オオシマトビケラ・カワニナ (B)・ゲンジボタル・コオニヤンマ・コガタシマトビケラ・スジエビ・ヒラタドロムシ・ヤマトシジミ
きたない水	イソコツブムシ・タイコウチ・タニシ・ニホンドロソコエビ・ヒル (A)・ミズカマキリ・ミズムシ
大変汚い水	アメリカザリガニ・エラミミズ・サカマキガイ (A)・セスジユスリカ (C)・チョウバエ

この表を見ると、出現する生物種から A と C に属する生物種は、きたない水と対応します。一方、B に属する生物種は、「きれい」あるいは「少し汚い」水である指標生物と対応します。A と C の生物の出現頻度が高い地点は、それぞれ、E 地点と 4 つの地点 (H, G, I ,J) です。

このように、F 地点を除く下流地域は、「きたない」あるいは「大変汚い」水質であることがわかります。一方、A-F 地点は、B グループに属する生物種の出現頻度が高い。これらは全て上流地域です。

単に上流から下流にかけて水質が悪くなるとすると、E 地点が「きたない」あるいは、「大変汚い」水質であり、F 地点では「きれい」あるいは「少し汚い」と判断されており、「上流から下流にかけて水質が悪くなる」ということのみでは説明できなくなります。なぜでしょうか。地形をみると E 地点で二つの川がつながっており、ここで水質が悪くなっているのかもしれません。もう一方の川の生物を調べることも考えるべきかもしれません。

水質の化学分析において、PO4（PO_4^{3-}[ppm]）、NH4（NH_4^+[ppm]）]は、4 地点（G-J）で高くなっています。また、有機物が多く水質が悪化した水ほど COD は高くなる傾向があります[文献を調べよう]。また、アンモニウム態窒素は、主としてし尿や家庭下水中の有機物の分解や工場排水に起因するもので、それらによる水質汚染の有力な指標となります[文献を調べよう]。さらに、リン酸の値が高いということは、生物の分解、生活排水の流れ込みなどによって生じた汚れが多いということと関係しています[文献を調べよう]。これらの指標が高い地域は、下流の G-J の 4 地点であり、特にこの 4 地点での水質汚染が高いと考えられます。

F 地点になると再度「きれい」あるいは「少し汚い」に変化しています。このような変化がおこる可能性としては、植物など生物による川の浄化メカニズムとも関係があるのかもしれません。この現象を説明する考察してデータを楽しんでみてください。

Pynarakita11

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.preprocessing import StandardScaler

# データの定義
sawa = [ 4,  3,  0,  2,  0,  0,  0,  0,  0,  0]
kage = [ 6,  9,  7,  4,  4,  6,  5,  2,  3,  0]
hebi = [ 0,  3,  0,  0, 15,  5,  0,  0,  0,  0]
tobi = [ 3,  5,  0,  2,  2,  3,  0,  0,  0,  0]
kawa = [ 4,  3,  0,  5,  0,  0,  0,  0,  0,  0]
hiru = [ 0,  0,  1,  0,  3,  0,  2,  1,  0,  0]
saka = [ 0,  0,  2,  0,  4,  0,  0,  0, 17,  0]
ito  = [ 0,  0,  0,  0,  0,  0,  3,  0,  7, 14]
yusu = [ 0,  0,  0,  0,  0,  0,  7,  6,  0, 13]
COD  = [ 5,  5, 10, 10, 20, 10, 20, 20, 10, 50]
PO4  = [0.02, 0.02, 0.02, 0.02, 0.05, 0.02, 0.1, 0.1, 0.1, 0.1]
NH4  = [np.nan, 0.2, 0.2, np.nan, 0.2, 0.2, 0.5, 0.5, 0.5, 0.5]

# データセットを作成し、行名を設定
dataSet = pd.DataFrame({
    'sawa': sawa,
```

```
    'kage': kage,
    'hebi': hebi,
    'tobi': tobi,
    'kawa': kawa,
    'hiru': hiru,
    'saka': saka,
    'ito' : ito,
    'yusu': yusu,
    'COD' : COD,
    'PO4' : PO4,
    'NH4' : NH4
}, index=['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J'])

# 標準化を行う
scaler = StandardScaler()
DS_scaled = scaler.fit_transform(dataSet.fillna(dataSet.mean()))

# ヒートマップを作成
sns.heatmap(DS_scaled, cmap='coolwarm', xticklabels=dataSet.columns,
yticklabels=dataSet.index)
plt.title('Heatmap of Scaled Data')
plt.show()
```

文献

[1] 環境省水・大気環境局と国土交通省河川局による子供向けの調査手引き書『川の生きものをしらべよう』(発行;日本水環境学会・2006)

高校生からのコメント(思考実験)

中学生のとき、植生調査をした経験から、植生をてがかりとした、農作物の最適配置の可能性を思いついた。まず、栽培されている作物とその近くに生育する野草の種類の関係のマップを作成する。これにより、野草植生ごとの土を採取し、最適と予測される作物を栽培する。このように最適作物栽培についての土の提案ができるのではないかと考えた。日本の食料自給率の向上にも貢献できるかもしれない。

6.9 物理現象を説明する

「室内に放置された湯の温度が時間とともに低下する」という事象を数理モデルで表現しシミュレーションをすることを考えた。
モデル 1： 始めの湯の温度を T_0（℃）、一分当たりの湯の温度の低下を a（℃）とし、t 分後の温度 $T(t)$（℃）を、

$$T(\mathrm{t}) = T_0 - at$$

とする。
モデル 2： t 分後と $t+1$ 分後の温度の関係を

$$T(t+1) = T(t) - bT(t)$$

とする。
モデル 3： 室温 $\mathrm{T_{room}}$ を考慮したモデルを考えた。

$$T(t+1) = T(t) - c(T(t) - T_{room})$$

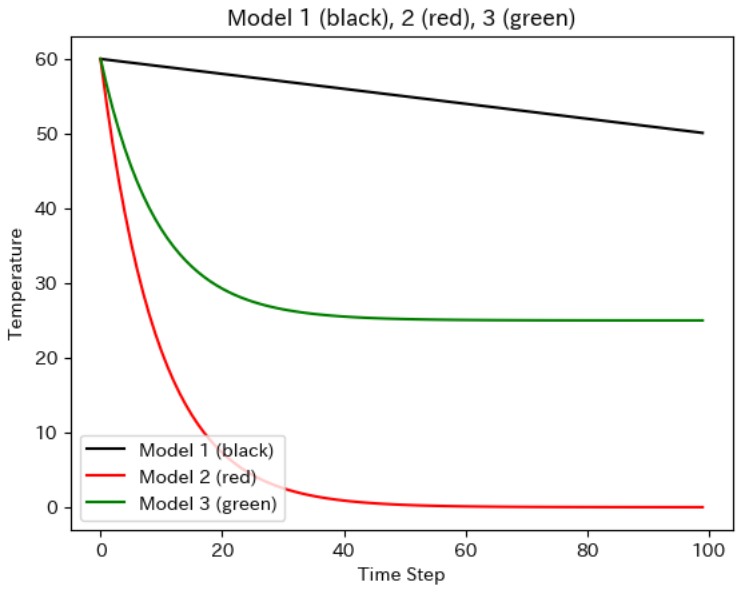
［啓林館、理数探究基礎：未来に向かって p.35］

数理モデル 1-3 を Pynarakita012 によりシミュレーションしてみよう。

Pynarakita012

```
import matplotlib.pyplot as plt
import numpy as np

# シミュレーションの設定
ns = 100

# [1] モデル 1 の温度低下を計算
T0 = 60
a = 0.1
T1 = np.zeros(ns)
T1[0] = T0
for i in range(1, ns):
    T1[i] = T0 - a * i

# [2] モデル 2 の温度低下を計算
T2 = np.zeros(ns)
T2[0] = T0
b = 0.1
for i in range(1, ns):
    T2[i] = T2[i-1] - b * T2[i-1]

# [3] モデル 3 の温度低下を計算
T3 = np.zeros(ns)
T3[0] = T0
c = 0.1
Troom = 25
```

```
for i in range(1, ns):
    T3[i] = T3[i-1] - c * (T3[i-1] - Troom)

# データフレームを作成してプロット
plt.plot(T1, label='Model 1 (black)', color='black')
plt.plot(T2, label='Model 2 (red)', color='red')
plt.plot(T3, label='Model 3 (green)', color='green')
plt.title('Model 1 (black), 2 (red), 3 (green)')
plt.xlabel('Time Step')
plt.ylabel('Temperature')
plt.legend()
plt.show()
```



6.10 物理現象を説明する

A 君は、「重りのついた糸の下側 b をはやく引っぱると上側の糸 a で切れ、ゆっくり引っ張ると下側の糸 b をきることができる（**図 2**）」。つまり、上（a）、下（b）のそれぞれの糸を、b の糸の末端をひっぱって、a で切ることも、b で切ることも自在にできるのだそうだ。これを数理モデルで説明してみましょう。これはたぶん、高校生には難しい課題だと思う。が、できないこともない。チャレンジしてみよう。

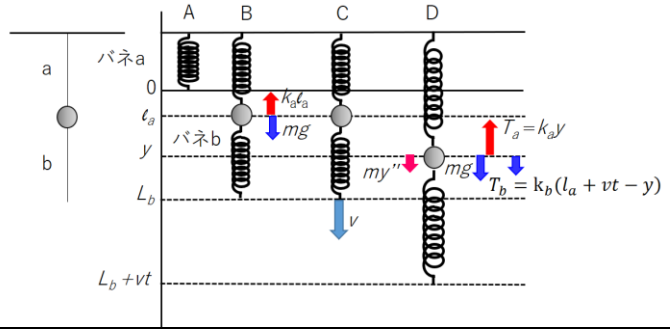


図 2 重りの二つの糸を自在に切る数理モデル
［啓林館、理数探究基礎:未来に向かって、p.86］

モデルの説明(Materials and Methods)

左側の重りと糸を、右側 B のように重りをバネで挟んだモデルで考えてみよう。なぜここでバネにするかというと、下側のバネの末端をもって、ぎゅっと引っ張る(これを速度 v)としよう。そして、このときの、二つのバネの張力(上バネの張力を T_a 、下のバネの張力を T_b)を比較して、$T_a > T_b$ のときバネ a の張力がバネ b に比べて大きいので、左側の重りと糸では a が切れることになる。一方、$T_a < T_b$ のときは、重りと糸では、b で切れるということになると仮定できます。この二つの状態が速度 v により定義することができるはずです。

背景調査

(a) バネモデル

重りとバネの問題は、高校の教科書でも説明されており、また、いろいろな書籍でも説明されている。バネののびと重りの運動方程式は、フックの法則「おもりにはたらく重力に比例し，弾性力の大きさはばねの自然長からの伸びに比例する」というものがある。これを使って重りが重力とつりあって止まっている状態(図 2A)を運動方程式で書いてみよう。

$$my'' = mg - k_a l_a \qquad (1)$$

重りの加速度を y'' とすると、my'' は重りにかかる力となる。いま、y はバネ a の末端を 0 と下側に座標を取った。ここで、m は重りの重さ、g は重力定数、k_a はバネ定数、l_a はバネの伸びです。

加速度 y'' とは $y'' = \frac{d^2}{dt^2}y = \frac{d^2y}{dt^2}$ などと書き、物体の位置 y における重りの加速度です。位置を 2 回微分すると加速度となります。いま、図 A では重りは止まっているので、$my'' = 0$ となり、

$$l_a = \frac{mg}{k_a} \qquad (2)$$

となるので、バネの伸び l_a を、重力 g ， 重りの質量 m とばね係数 k_a であらわらすことができます。

バネ係数の単位は、N/m です。1N は 1kg の質量を持つ物体に $1m/s^2$ の加速度を生じさせる力であり、1N/m は 1m バネを伸ばすのに 1N の力が必要である、ということを表しています。ちなみに、輪ゴムは 22N/m ぐらいだそうです。

速度 v で B のバネの下側を引っ張ると仮定しよう。いま、a のバネの末端を 0 として、下側に重りの位置を y とします。すると、重りの質量を m とすると、この重りに上向きにバネの張力 $T_a(= k_a y)$ 、下側に重力 mg 、また、下側のバネ b からの張力 T_b (= $[L_b + vt - y -$

$(L_b - l_a)] = vt - y + l_0$) となり、T_a は上向き、重力と T_b は下向きに働いて、重りは加速度 y'' で下側に動いているとしましょう。
すると運動方程式は、

$$my'' = mg - k_a y + k_b(vt - ly + l_a) \qquad (3)$$

となります。この式をもとに重りは動いていることになります。
バネ a が上向きに引っ張る力
　また、二つのバネの張力 T_a と T_b は、

$$T_a = k_a y$$

$$T_b = k_b(l_a + vt - y)$$

であるので、速度 v を変えたときに、$T_a > T_b$ と $T_b > T_a$ がはじめにおこるかどうかを確認してみればいいのだろう。と考えた。
　ここまでで、フックの法則など物理の知識で運動方程式をたてることができました。では、(3) の式を解くにはどうしたらいいだろうか。ここからが数学の調査です。

(b) 微分方程式を解く方法

　微分方程式がわからなくてもいいから、類似の形式の微分方程式を探してみよう。式 3 の微分方程式は、時間 t と座標 y で記述されています。そこで、y と t に注目して整理すると、

$$\frac{d^2y}{dt^2} + \frac{k_a+k_b}{m}y = g + \frac{k_b l_a}{m} + \frac{k_b v}{m}t \qquad (4)$$

となる。定数項を

$$c_1 = \frac{k_a+k_b}{m},\quad c_2 = g + \frac{k_b l_a}{m},\quad c_3 = \frac{k_b v}{m}$$

として式 4 を書き直すと、

$$\frac{d^2y}{dt^2} + c_1 y = c_2 + c_3 t \qquad (5)$$

と書ける。このような微分方程式の解法を文献[1]で調べてみると、この微分方程式の解は

$$y = a\cos(\sqrt{c_1}t) + b\sin(\sqrt{c_1}t) + \frac{c_2}{c_1} + \frac{c_3}{c_1} \qquad (6)$$

となることが解説されている。
$c_1,\ c_2,\ c_3$を元に戻すと、

$$y = a\cos\left(\sqrt{\frac{k_a+k_b}{m}}t\right) + b\sin\left(\sqrt{\frac{k_a+k_b}{m}}t\right) + \left(g + \frac{k_b l_a}{m}\right)\frac{m}{k_a+k_b} + \frac{k_b v}{k_a+k_b}t$$

さらに $mg = k_a l_a$ の関係を用いると

$$y = a\cos\left(\sqrt{\frac{k_a+k_b}{m}}t\right) + b\sin\left(\sqrt{\frac{k_a+k_b}{m}}t\right) + l_a + \frac{k_b v}{k_a+k_b}t$$

$y\,(t=0)$ のとき l_a であるので、

$$l_a = a + l_a \text{よって}$$

$$a = 0$$

$$y = b\sin\left(\sqrt{\frac{k_a+k_b}{m}}t\right) + l_a + \frac{k_b v}{k_a+k_b}t$$

また重りの速度は $t=0$ のとき $\frac{dy}{dt}=0$ であるので

$$\frac{dy}{dt} = -b\sqrt{\frac{k_a+k_b}{m}}\cos\left(\sqrt{\frac{k_a+k_b}{m}}t\right) + \frac{k_b v}{k_a+k_b}$$

に $t=0$ と $\frac{dy}{dt}=0$ を代入すると

$$b = -\frac{k_b v}{k_a+k_b}\sqrt{\frac{m}{k_a+k_b}}$$

最終的に、

$$y = -\frac{k_b v}{k_a+k_b}\sqrt{\frac{m}{k_a+k_b}}\sin\left(\sqrt{\frac{k_a+k_b}{m}}t\right) + l_a + \frac{k_b v}{k_a+k_b}t \qquad (7)$$

となります。

また、ここで求めた y を使って、時刻 t のバネ a とバネ b の張力は、

$$T_a = k_a y$$

$$T_b = k_b(l_a + vt - y)$$

により求めることができます。

余談 1

式 5 から式 6 の誘導

式 (5) の左辺は、$D^2y = \frac{d^2y}{dt^2}$ 、$Dy = \frac{dy}{dt}$ (この項は式 5 にない)とすると、

$$\frac{d^2y}{dt^2} + \mathrm{c}_1 y = (D^2 + D)y$$

とかける。D に注目した特性多項式つまり、$D^2 = \lambda^2$ 、$D = \lambda$ として式をつくり 0 とおく。

$$\lambda^2 + k = 0$$

これを解くと、$\mathrm{c}_1 > 0$ であるので、

$$\lambda = 0 \pm \sqrt{\mathrm{c}_1}i$$

となる。ここで、i は虚数であり、$i^2 = -1$ である。確かに、$\sqrt{\mathrm{c}_1}i \cdot \sqrt{\mathrm{c}_1}i = ki^2 = -\mathrm{c}_1$ となる。

このとき、

$$y = e^{0x}\{a\cos(\sqrt{c_1}t) + b\sin(\sqrt{c_1}t)\}$$

となる。
いま、$e^{0x} = e^0 = 1$となるので、

$$y = a\cos(\sqrt{c_1}t) + b\sin(\sqrt{c_1}t)$$

となる。
ここで、a と b は定数である。この式に $c_2 + c_3t$ と対応した一次の項 $d + et$ を

$$y = a\cos(\sqrt{c_1}t) + b\sin(\sqrt{c_1}t) + d + et \qquad (4)$$

がこの微分方程式の特殊解となる。
まず、定数 d と e を決めよう。それには、式 4 を式 3 に代入する。

$$\frac{dy}{dt} = -a\sqrt{c_1}\sin(\sqrt{c_1}t) + b\sqrt{c_1}\cos(\sqrt{c_1}t) + e$$

$$\frac{d^2y}{dt^2} = -ac_1\cos(\sqrt{c_1}t) - bc_1\sin(\sqrt{c_1}t)$$

となるので、式 3 の左辺は、

$$\frac{d^2y}{dt^2} + c_1y = -ac_1\cos(\sqrt{c_1}t) - bc_1\sin(\sqrt{c_1}t) + c_1\{a\cos(\sqrt{c_1}t) + b\sin(\sqrt{c_1}t) + d + et\}$$

$$c_1d + c_1et = c_2 + c_3t$$

t の一次式の係数を比較すると

$$d = \frac{c_2}{c_1}$$

$$e = \frac{c_3}{c_1}$$

となるので、

$$y = a\cos(\sqrt{c_1}t) + b\sin(\sqrt{c_1}t) + \frac{c_2}{c_1} + \frac{c_3}{c_1}$$

コンピュータ実験

式 7 をもとに Python プログラミングにより、定数値をいろいろ変えて計算機シミュレーションをしてみよう。とりあえず必要な式は、

$$y = -\frac{k_bv}{k_a+k_b}\sqrt{\frac{m}{k_a+k_b}}\sin\left(\sqrt{\frac{k_a+k_b}{m}}t\right) + l_a + \frac{k_bv}{k_a+k_b}t \qquad (7)$$

$$T_a = k_ay \qquad (8)$$

$$T_b = k_b(l_a + vt - y) \qquad (9)$$

です。Pynarakita01 は、これらの式を活用したプログラム(Python スクリプト)です。

Pynarakita01

```
import matplotlib.pyplot as plt
import numpy as np

# [1] 物理パラメータの設定
g = 9.8
m = 0.1
ka = 0.1
kb = 0.1

# [2] v の設定
v = 100

# [3] モデルパラメータの計算
mg = m * g
la = mg / ka
cs = (-kb * v / (ka + kb)) * np.sqrt(m / (ka + kb))
cp = np.sqrt(((ka + kb) / m))

# [4] Ta と Tb のシミュレーション
ns = 101
t = np.linspace(0, 10, ns + 1)
Ta = np.zeros(ns + 1)
Tb = np.zeros(ns + 1)
y = np.zeros(ns + 1)

for i in range(ns + 1):
    y[i] = cs * np.sin(cp * t[i]) + la + kb * v / (ka + kb) * t[i]
    Ta[i] = ka * y[i]
    Tb[i] = kb * (la + v * t[i] - y[i])

# [5] プロット 1
vset = np.concatenate([Ta, Tb])
minv = np.min(vset)
maxv = np.max(vset)
title1 = f"v={v}(m/s)"
plt.plot(Ta, Tb, 'b-', marker='o')
plt.xlabel("Ta(N)")
plt.ylabel("Tb(N)")
plt.title(title1)
plt.xlim(minv, maxv)
plt.ylim(minv, maxv)
plt.show()

# [6]
title2 = f"v={v}(m/s), Ta(black), Tb(red)"
plt.plot(t, Ta, 'k', label='Ta')
plt.plot(t, Tb, 'r', label='Tb')
```

```
plt.xlabel("Time")
plt.ylabel("N")
plt.title(title2)
plt.legend()
plt.show()
```

[1] 物理パラメータの設定

まず重力加速度(g=9.8 m/s²)、重りの重さ(m=0.1 kg)、バネ定数(ka=0.1 N/m, kb=0.1 N/m)と定義します。Python スクリプトでは、「=」は変数に値を代入することを示しています。ちなみに、輪ゴムのバネ定数はおおよそ 15-20 ぐらいだそうです。

[2] v の設定

下側の糸を引っ張る速度を定義します。v=100 とすると 100 m/s の速度で引っ張ることになります。

[3] モデルパラメータの計算

```
mg = m * g
```

m*g の値はいろいろなところで使うので、mg として計算しました。

```
la = mg / ka
```

重りをぶら下げつりあったところのバネの伸びを la としました。

```
cs = (-kb * v / (ka + kb)) * np.sqrt(m / (ka + kb))
```

は式 7 で重りの座標 y を計算するために必要な定数です。

[4] Ta と Tb のシミュレーション

```
t = np.linspace(0, 10, ns + 1)
Ta = np.zeros(ns + 1)
Tb = np.zeros(ns + 1)
y = np.zeros(ns + 1)
```

バネ b の末端を引っ張ったとき、0 から 0.01 秒での 11 点を(ns+10)についての時刻を t[1],…,t[11]、バネ a の張力を Ta[1],…,Ta[11]、バネ b の張力を Tb[1],…,Tb[11]、重りの座標 y[1],…,y[11]の空ベクトルを作成しました。

```
for i in range(ns + 1):
    y[i] = cs * np.sin(cp * t[i]) + la + kb / (ka + kb) * v * t[i]
    Ta[i] = ka * y[i]
    Tb[i] = kb * (la + v * t[i] - y[i])
```

つづいて、i を 0,1,2,…,10 として、y[1],…,y[11], Ta[1],…,Ta[11], Tb[1],…,Tb[11]を式 7-9 に従って計算しました。

[5] プロット 1

```
vset = np.concatenate([Ta, Tb])
minv = np.min(vset)
maxv = np.max(vset)
```

Ta と Tb を 2 次元にプロットします。x 軸と y 軸の範囲を Ta[1],…,Ta[10]、Tb[1],…,Tb[10]の 22 個の値についての最小値と最大値としました。

```
title1 = f"v={v}(m/s)"
plt.plot(Ta, Tb, 'b', marker='o')
plt.xlabel("Ta(N)")
plt.ylabel("Tb(N)")
plt.title(title1)
plt.xlim(minv, maxv)
plt.ylim(minv, maxv)
plt.show()
```

plot 関数を用いて、Ta と Tb の値をプロットしました。

#[6]
二つのバネの張力の時間変化

```
title2 = f"v={v}(m/s), Ta(black), Tb(red)"
plt.plot(t, Ta, 'k-', label='Ta')
plt.plot(t, Tb, 'r-', label='Tb')
plt.xlabel("Time")
plt.ylabel("N")
plt.title(title2)
plt.legend()
plt.show()
```

結果と考察

初期 0-0.01 秒の 11 点について、バネモデルにより v=1, 10, 100, 1000(m/s)でバネ b の下側を引っ張った時の二つのバネ a と b の張力を**図 1** に示します。図 1 では、$v = 1$ m/s のとき $T_a > T_b$ となり、バネ a の張力がバネ b に比べて大きいことがわかります。このことは、$v = 10$ m/s でも同様です。ところが、$v = 1000$ m/s となると、$T_b > T_a$ となります。張力の時間変化(**図 2**)でも同様のことが観察できます。ここで黒線は T_a、赤線は T_b を示しています。

これらのことを A 君の実験にあてはめて考えると、1N で糸が切れるとすると、A 君は、1-10m/s で糸を引っ張れば、重りの下で切れ、100-1000m/s で糸を引っ張れば、重りの上で切れるということになります。

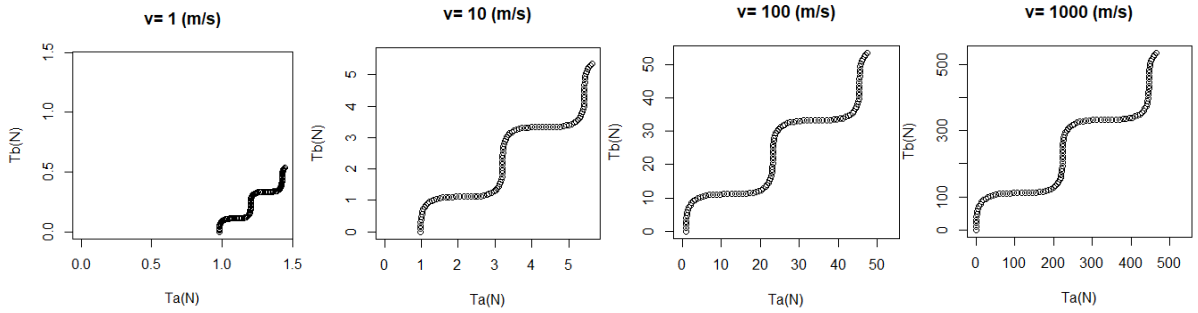
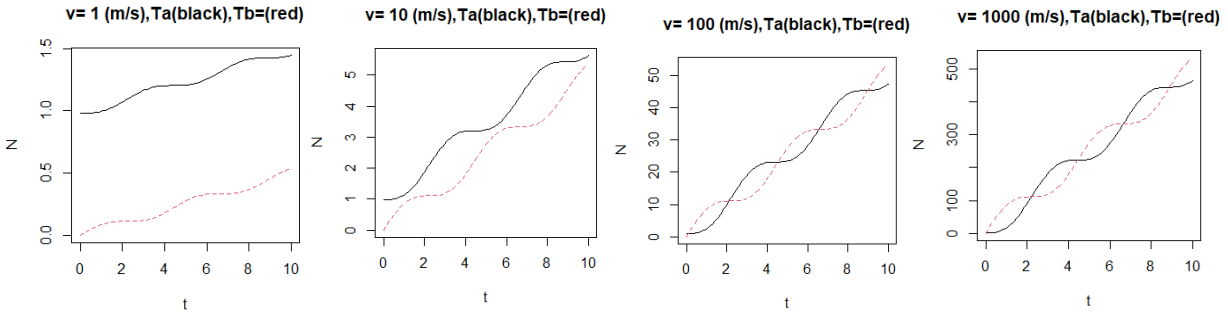


図 1 v=1, 10, 100, 1000(m/s)でバネ b の下側を引っ張った時の二つのバネの張力



今後の課題

A 君は、「重りのついた糸の下側 b をはやく引っぱると上側の糸 a で切れ、ゆっくり引っ張ると下側の糸 b をきることができる」ことは、バネモデルでシミュレーションすることができた。このようなバネモデルで説明できる物理現象をさらに検討してみたい。

なお、物理現象を微分方程式で記述し、この方程式を解き、プログラミングに実装してみると、視覚的に物理現象をとらえることができるので、さまざまな物理現象を検討するのもありでしょう。

文献

[1] 矢野健太郎著、微分方程式 裳華房(1980):同様の内容の本は多数出版されている。

6.11 ハッピー数

各桁の数の 2 乗和をとり、この操作により得られた自然数に同様の操作を繰り返していくと 1 になる自然数をハッピー数といいます。3 桁の自然数についてハッピー数かどうか検出するプログラムを作ってみよう。

［啓林館、理数探究基礎:未来に向かって、p.110］

こうゆう課題をみせられるとついつい解きたくなるのが研究者の悪いところです。とりあえずプログラムを作成しました。お楽しみください。とりあえず 10 桁の整数ではできそうです(Pynarakita24)。

1→1
2→4→16 →37→58→89→145→42→20→4
3→9→81→65→61→37→58→89→145→42→20→4
4→16 →37→58→89→145→42→20→4
5→25→29→85→89→145→42→20→4
6→36→45→41→17→50→25→29→85→89→145→42→20→4
7→49→97→130→10→1
8→64→52→29→85→89→145→42→20→4
9→65→61→37→58→89→145→42→20→4
となることを下線の数字を以下のプログラムの a に代入し確認した。

```
a = 81

# a が 9 より大きい間、以下の処理を続ける
while a > 9:
    sum = 0
    # 各桁の 2 乗和を求める
    while a != 0:  # a が 0 になるまで繰り返す
        sum += (a % 10) ** 2  # a を 10 で割った余りを 2 乗し sum に加算
        a //= 10  # a を更新
    a = sum  # 各桁の 2 乗和を a に代入
    print(a)  # a を出力
```

面白いところは、1→1 また 4 は 4 に戻れる。これを 4→→4 と書くと、2→→4、3→→4、5→→4、6→→4、7→→1、8→→4， 9→→4 となるので、最終的には 1 あるいは 4 になることがわかるので、Pynarakita24 では、終了条件は、4 か 1 となります。そこで、この終了条件でプログラムを作り変えたのが Pynarakita24 です。

Pynarakita24

```
X = 9999999999
a = X

# a が 1 でも 4 でもない間、以下の処理を続ける
while a != 1 and a != 4:
    sumv = 0
    # 各桁の 2 乗和を求める
    while a != 0:  # a が 0 になるまで繰り返す
        sumv += (a % 10) ** 2  # a を 10 で割った余りを 2 乗し sum に加算
        a //= 10  # a を更新
    a = sumv  # 各桁の 2 乗和を a に代入
    print(a)  # a を出力

# [2] 条件によるコメントの出力
if a == 1:
    comment1 = "You are happy!"
else:
    comment1 = "You are unhappy!"

comment1
```

9999999999 と代入すると
→810→65→61→37→58→89→145→42→20→4
と 4 に行きつきます。

```
> comment1
[1] "You are unhappy!"
```

ちっ、unhappy かいな！というふうに遊べます。
100 万までの数について、最終的に 1 と 4 となる数の総数をもとめると、

```
> table(X)
X
     1      4
143071 856929
```

となり、1 となるのは 14.3%となりました。(100 万より大きい数字についてはどうなるか不明)
14.3%は happy、85.7%は unhappy!

実際に以下のプログラムを動かしてハッピー数とアンハッピー数をカウントしてみよう。

Pynarakita26

```
import numpy as np

# ns の設定
ns = 1000000
X = np.zeros(ns, dtype=int)

# ns 回の繰り返し処理
for s in range(1, ns+1):
    a = s
    # a が 1 または 4 になるまでループ
    while a != 1 and a != 4:
        # a の各桁を取得し、それぞれを 2 乗して合計
        a = sum(int(digit) ** 2 for digit in str(a))
    X[s-1] = a

# X の要素の個数をカウント（ハッピー数とアンハッピー数のカウント）
Unique, counts = np.unique(X, return_counts= True)
aa = dict(zip(unique, counts))
aa
```

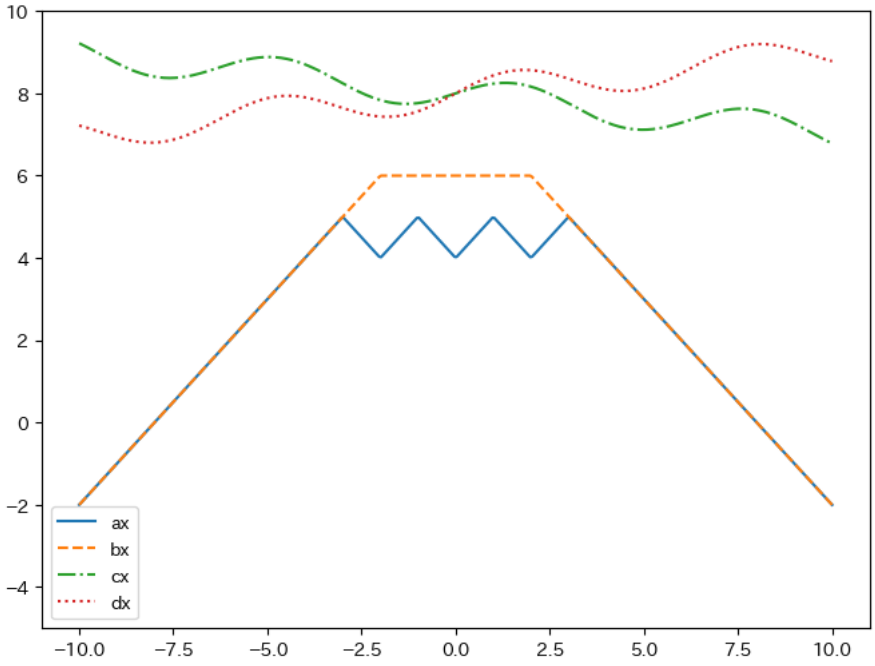
6.12 関数グラフアート

関数グラフアートというものがあるようで、関数 $y = f(x)$ となる関数をつくり、x に対して必ず y があるという関数を使って富士山を描いてみましょう。

［啓林館、理数探究基礎:未来に向かって、p.111］

Pynarakita25

```
import matplotlib.pyplot as plt
import numpy as np

# 関数の定義
# ax 関数: 複数の絶対値操作を行い、特定の形状を生成

def ax(x):
    return -np.abs(np.abs(np.abs(np.abs(x) - 1) - 1) - 1) + 5

# bx 関数: x が-2 から 2 の範囲内にある場合は 6 を、それ以外は x から 8 を引いた値の絶対値を返す
def bx(x):
    return np.where((-2 < x) & (x < 2), 6, -np.abs(x) + 8)

# cx 関数: sin 関数に線形項の絶対値を加えた値を返す
def cx(x):
    return 0.4 * np.sin(x) + np.abs(8 - 0.1 * x)

# dx 関数: sin 関数に線形項の絶対値を加え、x が増加するにつれて大きな値を返す
def dx(x):
    return 0.4 * np.sin(x) + np.abs(8 + 0.1 * x)

# x の範囲
x = np.linspace(-10, 10, 400)

# 各関数のプロット
plt.figure(figsize=(8, 6))
plt.plot(x, ax(x), label='ax')
plt.plot(x, bx(x), label='bx', linestyle='--')
plt.plot(x, cx(x), label='cx', linestyle='-.')
plt.plot(x, dx(x), label='dx', linestyle=':')

# y 軸の範囲設定
plt.ylim(-5, 10)

# 凡例の表示
plt.legend()

# グラフの表示
plt.show()
```


10
8
6
4
2
0
−2
−4
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
ax
bx
cx
dx

7. おわりに、能動的に学び世界に羽ばたこう

このテキストを読んで、とりあえず、学術論文の基本構成をもとに探究の仕方が理解できたと思います。また、集めたデータを客観的に評価する方法として統計学を中心に解説しました。インターネット等を通して膨大な知識や情報に触れることが可能になりました。このような知識が獲得できる現在、この知識を活用し、さらにいろいろな探究ができると思います。

また、いろいろ観察してみることから探究ははじまります。そこで、漠然とした探究目標をたてます。そして、探究目標をいくつかの課題にわけて、解けるものから解いていきます。難しい探究目標を立てる必要はありません。皆さんが興味のある事柄を、具体的に解析できそうな課題にわけていくということを進めればOKです。解析結果に疑問がでたら、統計を活用して仮説検定をするのもアリでしょう。運動方程式などをつくり解くのもありです。また、高校の教科書、はたまた専門書、学術論文を参考にしながら、課題を解決していくのも OK です。課題設定については、友人・高校の先生・研究者・専門家に相談するのもありでしょう。ただし、まず、自分で試行錯誤をするのがいいかと思います。グループで役割を適に分担して探究をすすめるのもありです。

さらに、集めたデータを理解した上で、新たな発明につなげるのもありです。モノづくりへと展開できる探究というのも、とってもカッコイイ！

課題解決に行き詰まったら友人・高校の先生・研究者・専門家に相談するのもいいでしょう。金谷の経験では、偉い大先生に相談したら「やっても意味がない！」などといわれたこともあります。でも、「しら～～」と研究を進めたら海外から高く評価されたということもあります。まあ、ポジティブな意見に耳を傾けて探究を継続するのでいいのかと思います。ポジティブにものごとを考えるというのはとってもだいじなことだと思います。さらには、「未知の問題をどう解決するか」のてがかりを実感できればなによりです。「理数探究」という教科の中で、皆さんがテーマを設定し、さまざまな分野を実感し、楽しく実験し、データ解析から客観的に結論を導きましょう。各自が興味を持てることを科学探究しましょう。

また、受験勉強に向けて、どのように戦略をとり、目指す大学へ合格するかというのも一つの実験例であり、探究であると思います。「ただひたすら勉強するのはつらいと思います、しかし自分なりの戦略を立てて、相手を知り勉強をする」となると、少しは、辛くなくなるかもしれません。理数探究を通して、「課題をみつけ、それを解決する」という総合力を身につければ、将来、社会で活躍するようになったときの問題解決能力の基礎を養うこともできるだろうと思います。

さあはじめよう、探究！Let’s enjoy 探究！

“Imagination is more important than knowledge…” (Albert Einstein)

“Good luck!”

謝辞

本テキストの作成にあたり、奈良県立奈良北高等学校、辻本裕明先生、仲田千鶴先生、竹田浩一先生、中川雅俊先生には、多大なるご協力およびご助言を頂きました。深く感謝いたします。

付録 Google Colaboratory 使い方

Python をデータ解析に活用するには、どうすればいいのだろうか？ 個人の PC で Python が使えるように環境構築することもできるが、パッケージのインストールエラーなどにてこずるなどの障害があるので、本書ではブラウザで Python を実行できる Google Colaboratory を用います。

Google アカウントの取得

まずは以下のサイトから Google アカウントを作成してください。もうお持ちの方は飛ばしてください。

https://support.google.com/accounts/answer/27441?hl=ja

Google Colaboratory へのアクセス

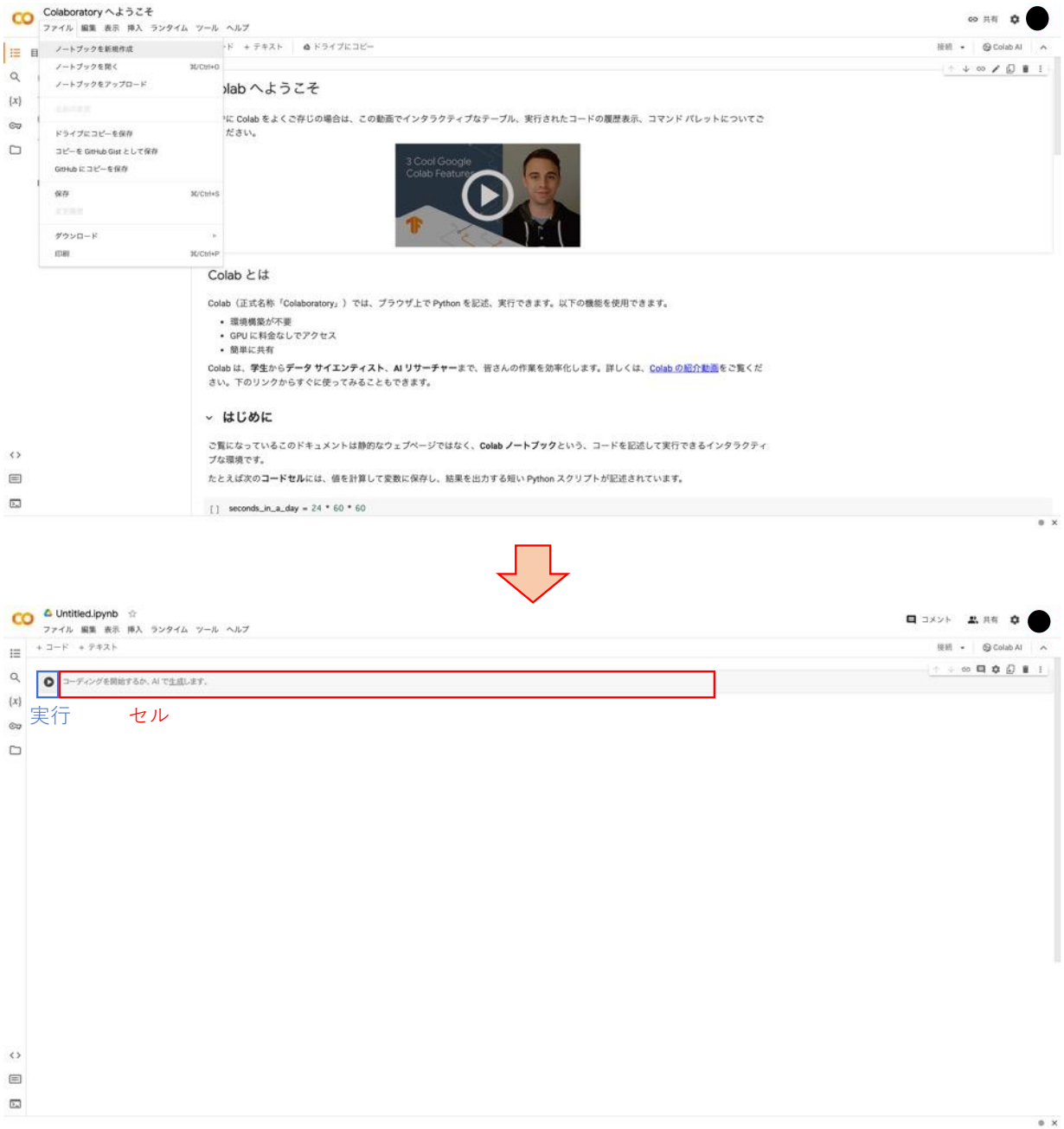
まずは、以下の URL にアクセスしましょう。

https://colab.research.google.com/notebooks/welcome.ipynb?hl=ja

そうすると、以下の画面が表示されるはずです。

Google Colaboratory でプログラムをつくる

下図に従って、「ファイル」から「ドライブの新しいノートブック」を選びます。そうすると何も書いてない画面が表示されます。プログラミングの準備が終わりました。赤で示しているセルという部分にコードを書き、青で示している▶ボタンでそのセルを実行できます。



では実際にプログラムを作成してみましょう。まずセルに

「1+2+3+4+5+6+7+8+9+10」

と入力します。この 1 行を選択し、上部の Run ボタンをクリックさせると、プログラムが実行されます（以下の図参照）その結果、セルの下に 1+2+3+4+5+6+7+8+9+10 を実行した結果である 55 が表示されています。

二次方程式 $ax^2 + bx + c = 0$ の解は、$x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$ です。これをもとに、二次方程式を解くプログラムを作成してみましょう。ここで、二次方程式について、a, b, c が分っているときの x を求めるプログラムは以下のようになります。

```
import numpy as np

a = 2
b = 7
c = 5

x1 = (-b + np.sqrt(b**2 - 4 * a * c)) / (2 * a)
x2 = (-b - np.sqrt(b**2 - 4 * a * c)) / (2 * a)

print("x1 : ", x1)
print("x2 : ", x2)
```

《 プログラムの概要 》

1 行目：numpy をインポートし、今後用いるときは np と書くことを宣言します。

3 行目：a = 2 とは、a に 2 を代入するという操作です。

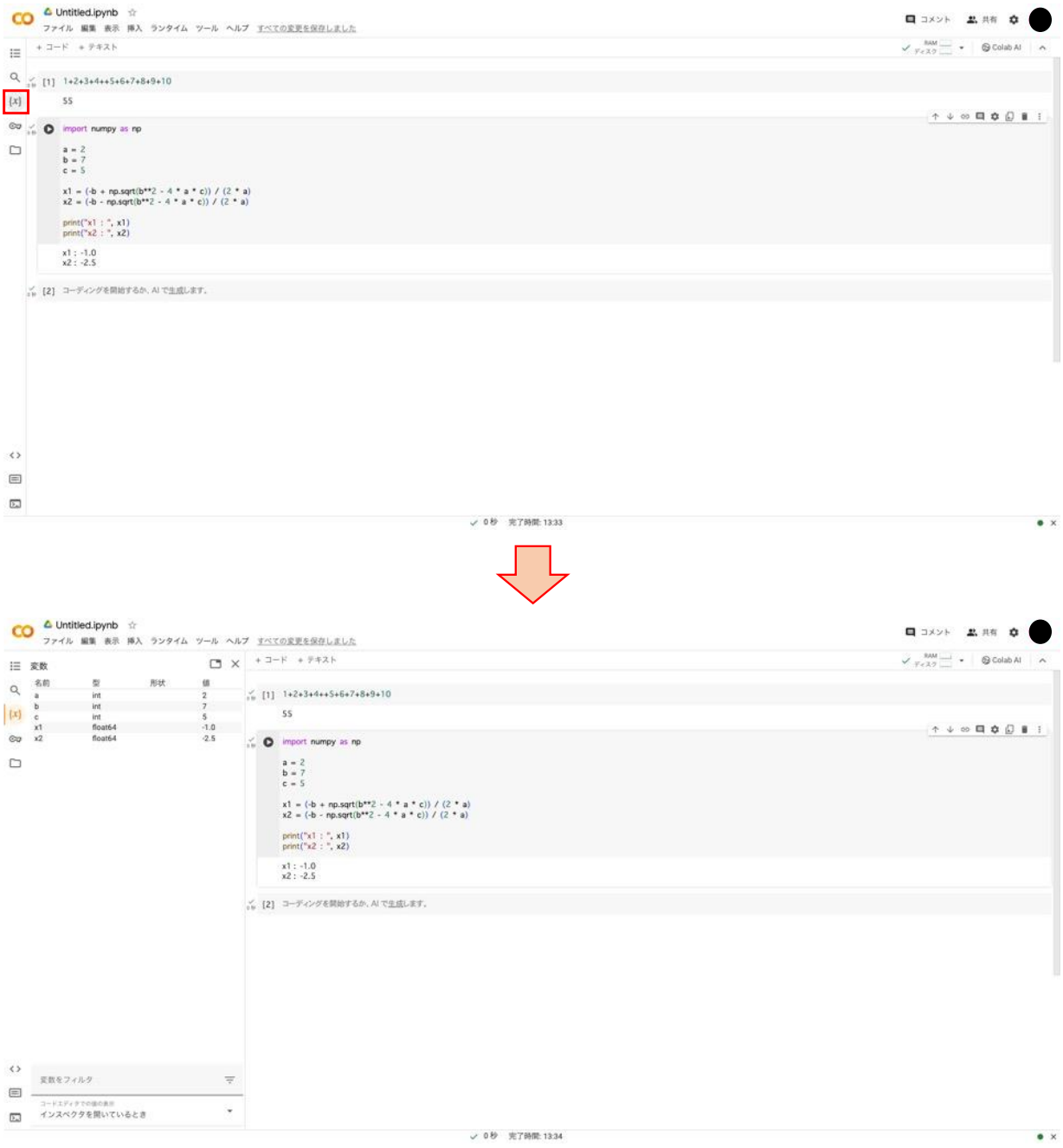
3-5 行目：a=2, b=7, c=5 が代入されました。

7 行目：$x1 = \frac{-b+\sqrt{b^2-4ac}}{2a}$

8 行目：$x2 = \frac{-b-\sqrt{b^2-4ac}}{2a}$

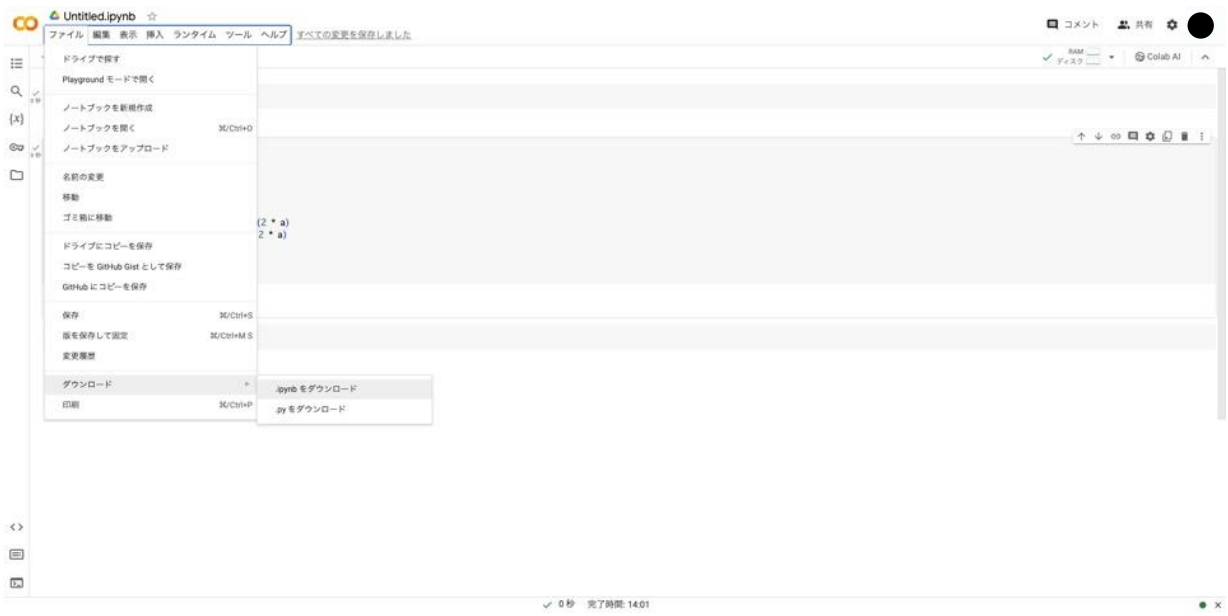
を Python により作成しました。ここで np.sqrt()は$\sqrt{\ }$の計算を行う関数です。

10,11 行目：x1 と x2 に格納された値（二次方程式の解）が出力されます。以下に実行結果を示しました。x1 と x2 を出力すると-1 と-2.5 となりました。$\{x\}$を押すと各変数とその型と値を確認することもできます。



Google Colaboratory で作成したプログラムを保存する

　ではこのプログラムをファイルにセーブしてみましょう。下図のようにメニューバーの「ファイル」から「ダウンロード」を選択し、「.ipynb をダウンロード」を選びダウンロードします。ダウンロードしたコードは個人の PC に環境を作れば、動かせますし、もう一度 Google Colaboratory にアップロードして動かすこともできます。



Google Colaboratory にプログラムをアップロードする

　下図のようにメニューバーの「ファイル」から「ノートブックをアップロード」を選択し、アップロードしたい ipynb ファイルをドラック&ドロップします。そうすることで、これまでに作ってきたプログラムを Google Colaboratory 上で動かせるようになります。


Colaboratory へようこそ
ファイル 編集 表示 挿入 ランタイム ツール ヘルプ
ノートブックを新規作成
ノートブックを開く ⌘/Ctrl+O
ノートブックをアップロード
ドライブにコピーを保存
コピーを GitHub Gist として保存
GitHub にコピーを保存
保存 ⌘/Ctrl+S
ダウンロード
印刷 ⌘/Ctrl+P
+ コード + テキスト
ドライブにコピー
共有
Colab AI
lab へようこそ
3 Cool Google Colab Features
Colab とは
Colab（正式名称「Colaboratory」）では、ブラウザ上で Python を記述、実行できます。以下の機能を使用できます。
• 環境構築が不要
• GPU に料金なしでアクセス
• 簡単に共有
Colab は、学生からデータ サイエンティスト、AI リサーチャーまで、皆さんの作業を効率化します。詳しくは、Colab の紹介動画をご覧ください。下のリンクからすぐに使ってみることもできます。
はじめに
ご覧になっているこのドキュメントは静的なウェブページではなく、Colab ノートブックという、コードを記述して実行できるインタラクティブな環境です。
たとえば次のコードセルには、値を計算して変数に保存し、結果を出力する短い Python スクリプトが記述されています。
[ ] seconds_in_a_day = 24 * 60 * 60


Colaboratory へようこそ
ファイル 編集 表示 挿入 ランタイム ツール ヘルプ
目次
はじめに
データ サイエンス
機械学習
その他のリソース
使用例
セクション
+ コード + テキスト
ドライブにコピー
共有
Colab AI
ノートブックを開く
例
最近
Google ドライブ
GitHub
アップロード
参照
または、ここにファイルをドラッグしてください
キャンセル
Colab とは
はじめに
seconds_in_a_day = 24 * 60 * 60

著者（執筆時所属）

張　凡　　奈良先端科学技術大学院大学・先端科学技術研究科・情報科学領域
今西　温輝　　甲陽学院高等学校、高校生
金谷　重彦　　奈良先端科学技術大学院大学・先端科学技術研究科・情報科学領域
松本　健一　　奈良先端科学技術大学院大学・先端科学技術研究科・情報科学領域
平尾　俊貴　　奈良先端科学技術大学院大学・先端科学技術研究科・情報科学領域
嶋利　一真　　奈良先端科学技術大学院大学・先端科学技術研究科・情報科学領域
工藤　拓斗　　奈良先端科学技術大学院大学・先端科学技術研究科・情報科学領域
田中　英武　　奈良先端科学技術大学院大学・先端科学技術研究科・情報科学領域
山﨑　和真　　奈良先端科学技術大学院大学・先端科学技術研究科・情報科学領域
德永　眞一郎　奈良先端科学技術大学院大学・戦略企画本部
三宅　雅人　　奈良先端科学技術大学院大学・研究推進機構
小笠原　司　　奈良先端科学技術大学院大学・地域共創推進室

データサイエンスで考える理数探究基礎
「未来に向かって」Python 版

2024 年 10 月 1 日　初版発行
編著者 金谷重彦
著作　奈良先端科学技術大学院大学
　　NAIST STELLA プログラム運営委員会
発行所 国立大学法人 奈良先端科学技術大学院大学
　　〒630-0192 奈良県生駒市高山町 8916-5
　　https://www.naist.jp/
　　電話 0743-72-5111（代表）

ISBN 978-4-902874-05-1