Hofstadterの蝶

志村昂輝

2026年1月8日

一級品として表れる諸々の技術や芸当は、その自由自在さで我々を驚かせますが、その背後には膨大なルールと型があります。研究も例外ではなく、よりよい研究で発揮される個性はまずある一定の型やルールを身に着けた後に生まれています。研究における型を身に着ける最初の手段は、例えば学部生のうちには標準的な教科書を読んで知識を増やすことでしょうが、これは全くもって序の口にすぎず、一人前となるためには配属された研究室で師弟関係を結び、その研究室が売りとするスタイルを吸収していかなければなりません。

ところで問題となるのが、この研究室の売りは一朝一夕に掴めるものではないということです。まず師弟関係の下で、弟子として師匠の謦咳に接しているかどうかが、素人とそうでない人とを分ける最初の分水嶺となります。将来自分の同業者になると見込んだ人間に対して、師匠はその分野の共通言語や常識を叩き込みますが、一人前とみなされるにはその量と厳しさをもってしても数年を要します。修士と博士合わせて 5 年かかるのは、この修行が並大抵のものではなく、素人から業界人へと決定的に変化させる重大なプロセスであることを示唆しているでしょう。

本稿は以上 2 つの困難を克服するために作成されています。つまり、加藤研内部で共有されている重要だがインアクセシブルな技術をまとめ、研究を始める際に典型的に出くわす困難とその対処法を示すこと、及び内容を極力端的に整理し、一学生から研究者になるための経路を効率よく整備することです。ただし、私の特殊なバックグラウンドに起因する限界について読者にお断りしなくてはなりません。まず加藤雄介研究室で扱われるテーマと技術は広く、私がここで示す内容はそのほんの一部にすぎないことです。もとより本研究室の強みは数値計算よりも微分方程式の求解や特殊関数の操作といった解析計算に存在しており、本稿の内容はむしろ傍流といえるかもしれません。主流といえるそちらの技術については、読者の皆様が教員との長い議論や共同研究の中で少しずつ身に着けたり、あるいは今後誰かが同様にドキュメント化してくれることを期待いたします。

第 1 章では、数値計算や解析の作業に適したプログラミング言語の紹介と簡単な文法の解説、及び環境構築の仕方について述べます。

強相関物質の研究スタイルにはかなり決まった型が存在していますが、出発点として死活的に重要なのがハミルトニアンの構築とその対角化です。第 2 章では、強束縛模型について軽く説明した後、ハミルトニアンを実装する方法、及びその対角化の方法について述べます。ハミルトニアンの対角化により得られた固有値がエネルギー分散ですが、実際に研究したり論文を書いたりするうえで必要なバンド図やフェルミ面の書き方についてもここで解説します。

第 3 章ではグリーン関数の実装方法について述べます。グリーン関数の虚部から得られる状態密度の描画方法もここで述べます。

第1章　プログラミング言語の選択と環境構築

1.1　数値計算向きの言語 (C++, Fortran)

数値計算という目的に絞ってもプログラミング言語の選択肢は多岐にわたりますが、物性物理で格子模型や多体問題を扱う場合、計算量は系の大きさや自由度の増加とともに急激に増大します。したがって、実行速度やメモリ配置を意識する必要があり、C++やFortranはそれにふさわしい選択肢の一つとして挙げられます。明示的に型が指定されるために、数値誤差のふるまいを把握しやすくなるのも重要です。

1.1.1　C++について

C++はC言語が基盤となっていますが、クラスやテンプレートといった機能を導入することで大規模なプログラムの構築や保守に向いています。本稿で主に用いる言語です。コンパイラもフリーで手に入り、特にLinuxでは標準装備されています。Windowsでもコンパイラがフリーで入手できるようです。

数値計算上一番大きなメリットは、実行速度の速さです。C++はコンパイル型言語で、ループなどが機械語に近い形で最適化されるために、行列演算や反復計算で高い性能を発揮します。また配列の確保や解放を動的に行えば巨大配列を扱うこともできるので、波数空間上の物理量の情報を格納する場合に非常に効率がよいです。数値計算のライブラリも充実しています。

デメリットとして、メモリ管理を意識した低水準な記述が前提となっているために、数値計算にバグが混じりやすくなることが挙げられます。Fortranに比べると覚える概念が多く、特にポインタは大部分の学習者が躓く場所として悪名高いもので、言語に対する正確な理解と注意深い実装が求められます。それでも私がここでC++をお勧めするのは、学習コストが高い分ほかの言語を学習する際のハードルが下がるであろうことや、Fortranに比べるとできることが多いこと、最後に身も蓋もありませんが、筆者がFortranよりも長く触れているためにドキュメントを作りやすいことがあります。例えば個人的に文字列操作はFortranよりもC++のほうが便利だと考えています。

1.1.2　Fortranについて

数値計算の言語としてはFortranも依然重要な選択肢といえます。物性理論だけでなく、科学技術計算に特化した言語として長い歴史を持つために、研究室によっては保守性

の観点から Fortran によるコーディングを強いる場合もあります。これはオリジナリティ保護の観点から理にかなっていて、研究室の強みが Fortran に支えられており、いわば研究室独自のライブラリとして使いまわすことができるのです。大規模プロジェクトならなおさら、既存の技術的な資産を一人で書き換えることも不可能でしょう。しかしながら加藤雄介研究室は数値計算の研究室ではなく、そのような蓄積はないので絶対に Fortran を使わなければいけない環境ではありません。

Fortran にはポインタなどの概念が存在せず、比較的コーディングしやすいのも利点です。数値計算以外には基本的に向いていないのがデメリットで、Fortran ができるよりも C++ができる人のほうが、今後数値計算以外のことを仕事にする場合に融通が利くのではないかと考えています。

1.2 補助としての言語 (Python)

実のところ筆者が一番長く触れている言語です。数値計算を補助する言語として Python3 も非常に有用です。C++や Fortran に比べると、Python3 はライブラリが充実していて、可視化やデータ解析を迅速に行える利点もあります。[1] 型指定もないので、簡単な計算を行いたいだけなら C++や Fortran よりも圧倒的に便利です。特に配列 (リスト) の定義が楽で、線形代数関連のライブラリが充実しており簡単に計算できるのは大きな魅力です。コードの読解も比較的やさしく、精神的な負担が少ないです。

Python3 はスクリプト言語であり、実行速度やメモリ制御の面ではコンパイル型言語に劣るために、大規模数値計算の中核を担うのには不適切である場合もあります[2]。また型宣言がないのは実装や保守性の観点から苦しみの種となる場合もあります。しかしながら、C++や Fortran で実行した本計算の出力を Python で読み込み、プロットや解析を行うといった使い方は広く行われており、その自由度はほかの言語の追随を許しません。

1.3 近年注目されている言語 (Julia)

筆者は常用しておらず、軽く触れる程度しかできませんが、近年数値計算を目的としたプログラミング言語として注目されているものの一つに Julia があります。素早く実装することができ、かつ可読性が高い部分は Python に似ていますが、型安定なコードなら C や Fortran に劣らない速度で計算できるため、両者のいいとこどりをしています。現時点でのデメリットといえば、比較的新しい言語であるためにライブラリやパッケージの進化が早くバージョン管理が大変であろうことだと考えられますが、最近は研究会なども開かれており今後利用者は増えていくと見込まれます。

[1] 可視化には matplotlib, 数値計算やデータ解析には numpy や scipy といったライブラリが便利でしばしば使われます。

[2] 厳密には計算効率を上げるための工夫が様々に存在するようですが、その工夫に割くコストを考えると最初から C++や Fortran で実装した方がよいという意見が専門家の間では散見されます。

1.4　C++で使える高速数値計算ライブラリ

関連図書

[1] Hofstadter, Douglas R., Energy levels and wave functions of Bloch electrons in rational and irrational magnetic fields, Phys. Rev. B **14**, 2239(1976).

[2] C R Dean 1, L Wang, P Maher, C Forsythe, F Ghahari, Y Gao, J Katoch, M Ishigami, P Moon, M Koshino, T Taniguchi, K Watanabe, K L Shepard, J Hone, P Kim, Hofstadter's butterfly and the fractal quantum Hall effect in moiré superlattices, Nature 497(7451), (2013)