

目次

口絵	iii
翻訳にあたって	v
序	vii
記号解説	xiii

第 1 章 科学技術計算のための Python への入門	1
1.1 インストールとセットアップ	3
1.2 Numpy	5
1.2.1 Numpy の配列とメモリ	7
1.2.2 Numpy の行列	11
1.2.3 Numpy のブロードキャスト	12
1.2.4 Numpy のマスクされた配列	14
1.2.5 Numpy の最適化と内容見本	15
1.3 Matplotlib	16
1.3.1 Matplotlib の代替	18
1.3.2 Matplotlib の拡張	19
1.4 IPython	19
1.4.1 IPython Notebook	21
1.5 Scipy	23
1.6 Pandas	24
1.6.1 シリーズ (Series)	24
1.6.2 データフレーム (Dataframe)	26
1.7 Sympy	29
1.8 コンパイル済みライブラリのインタフェース	32
1.9 統合開発環境	33
1.10 パフォーマンスと並列プログラミングへのクイックガイド	33
1.11 その他のリソース	37
参考文献	38

第2章 確率	39
2.1 はじめに	39
2.1.1 確率密度の理解	40
2.1.2 確率変数	41
2.1.3 連続型確率変数	47
2.1.4 微分積分を超えた変数の変換	50
2.1.5 独立確率変数	52
2.1.6 折れた竿の古典的事例	54
2.2 写像法	55
2.2.1 重み付きの距離	58
2.3 写像としての条件付き期待値	59
2.3.1 付録	65
2.4 条件付き期待値と平均二乗誤差	66
2.5 条件付き期待値と平均二乗誤差最適化の実施例	69
2.5.1 実施例	70
2.5.2 実施例	73
2.5.3 実施例	76
2.5.4 実施例	80
2.5.5 実施例	81
2.5.6 実施例	83
2.6 情報エントロピー	85
2.6.1 情報理論の概念	85
2.6.2 情報エントロピーの性質	88
2.6.3 カルバック・ライブラー情報量	89
2.7 積率母関数	90
2.8 モンテカルロサンプリング法	93
2.8.1 離散型変数のための逆 CDF 法	94
2.8.2 連続変数のための逆 CDF 法	96
2.8.3 棄却法	98
2.9 有用な不等式	102
2.9.1 マルコフの不等式	102
2.9.2 チェビシエフの不等式	103
2.9.3 ヘフディングの不等式	105
参考文献	106

第3章 統計	107
3.1 はじめに	107
3.2 統計用 Python モジュール	108
3.2.1 Scipy の統計モジュール	108
3.2.2 Sympy の統計モジュール	109
3.2.3 その他の統計用 Python モジュール	110
3.3 収束の種類	110
3.3.1 ほとんど確実に収束	111
3.3.2 確率収束	113
3.3.3 分布収束	116
3.3.4 極限定理	116
3.4 最尤推定法を用いた推定	118
3.4.1 コイン投げ施行の準備	120
3.4.2 デルタ法	129
3.5 仮説検定と P 値	132
3.5.1 コイン投げの例に戻る	133
3.5.2 ROC (受信者動作特性)	137
3.5.3 P 値	139
3.5.4 検定統計量	140
3.5.5 多重仮説検定	147
3.6 信頼区間	148
3.7 線形回帰	152
3.7.1 多重共変量への拡張	162
3.8 最大事後確率	167
3.9 ロバスト統計	173
3.10 ブートストラッピング	180
3.10.1 パラメトリックブートストラップ	185
3.11 ガウス = マルコフの定理	186
3.12 ノンパラメトリック法	190
3.12.1 カーネル密度推定	190
3.12.2 カーネル平滑化	193
3.12.3 ノンパラメトリック回帰推定量	198
3.12.4 最近傍回帰	199
3.12.5 カーネル回帰	203

3.12.6 次元の呪い	205
参考文献	207
第4章 機械学習	209
4.1 はじめに	209
4.2 Pythonの機械学習モジュール	209
4.3 学習の理論	214
4.3.1 機械学習理論への入門	215
4.3.2 汎化の理論	220
4.3.3 汎化と近似の複雑さについての動作例	222
4.3.4 交差検定	228
4.3.5 バイアスとバリエーション	233
4.3.6 学習ノイズ	236
4.4 決定木	239
4.4.1 ランダムフォレスト	246
4.5 ロジスティック回帰	248
4.5.1 一般化線形モデル	253
4.6 正則化	254
4.6.1 リッジ回帰	258
4.6.2 Lasso 回帰	263
4.7 サポートベクトルマシン	265
4.7.1 カーネルトリック	269
4.8 次元削減	270
4.8.1 独立成分分析	274
4.9 クラスタリング	279
4.10 アンサンブル手法	283
4.10.1 バギング	283
4.10.2 ブースティング	285
参考文献	287
索引	289
参考文献リスト (和書)	296

第1章

科学技術計算のための Python への入門

Getting Started with Scientific Python

Python は、ここ数年主流となってきた。いまや、Python は工学およびコンピュータサイエンスにおける多くの学部カリキュラムの一部となっている。優れた書籍および対話型のオンラインチュートリアルを見つけることは簡単である。特に、Python は、Django や CherryPy などのフレームワークを用いたウェブプログラミングの分野で十分な地位を確立しており、これらはアクセス数の多い多くのサイトの基盤となっているプラットフォームである。

ウェブプログラミングを越えて、線形代数から機械学習の可視化にいたる多くの科学分野全体にわたって、サードパーティによる拡張機能のリストが今も増え続けている。これらのアプリケーションにおいて、Python はメソッドの交換や、Fortran や C で書かれる典型的なコアルーチンに対してのデータを引き渡しを簡単にするための、いわばグルー（糊付け）としての役割を果たすソフトウェアである。科学技術計算のための Python は、ほぼ 20 年にわたり、産官学で基本的なものであり続けている。たとえば、米航空宇宙局 (NASA) のジェット推進研究所は、宇宙船の軌道の計画と可視化のために Fortran/C++ ライブラリのインタフェースとして科学技術計算のための Python を使用している。また、ローレンス・リバモア国立研究所は、いくつかの定型テキスト処理や、ビッグデータの先進的な可視化 (VISIT など^[1]) など、多種多様な計算タスクに科学技術計算のための Python を利用している。シェルリサーチ、ボーイング、インダストリアル・ライト & マジック、ソニー・エンタテインメント、および P&G は、日々のデータ処理や分析に科学技術計算のための Python を使用している。Python は、このように十分に確立され、様々な分野で拡張し続けている。

Python は正式なソフトウェア開発のトレーニングを受けていない科学者やエンジニア向けの言語である。Python は、何の障害もなく、プロトタイプや設計、シミュレーション、およびテストを行うために利用される。これは Python が、本質的な容易さや、反復型開発サイクル、既存コードの相互運用性、信頼性の高いオープンソースコードの巨大な基盤へのアクセス、および階層的に区画化された設計理念を提供するためである。Python の生産性は、ユーザのワークフロー (実行時間とプログラミング時間など) に強く影響されることが知られている^[2]。このため、