

WBIC を用いた PyMC コミット履歴の統計的分析

杉山 透
放送大学
宇部工業高等専門学校
1520390580@campus.ouj.ac.jp

中谷 多哉子
放送大学
tinakatani@ouj.ac.jp

要旨

本稿では、競合 OSS 登場への対応や大小様々なバージョンアップなど、多様な進化を経験した OSS である PyMC におけるコミット回数がどのような統計モデルに従うかを明らかにする。この問題に対して、複数の統計モデルを用意し、WBIC を指標としたモデル選択を行うことでアプローチする。そして、モデル選択の結果からコミット回数分布の性質を解釈し、OSS 開発におけるコミット増加や開発者参入・離脱の仕組みを考察する。

解析の結果、PyMC のコミット履歴が Zipf 分布に最も適合することが明らかになった。これは、コミット回数の増加が人口増加を説明する Simon モデル、またはジブラ過程に何らかの変形を加えたモデルを OSS におけるコミット増加に援用したモデルによって生じた可能性を示唆する。今後は、より詳細な解析を行うことで、Simon モデルや変形を加えたジブラ過程モデルなど、どのようなモデルから Zipf 分布的なコミット履歴が生成されたか明らかにしていく。

1. はじめに

これまで、ソフトウェア進化 [1] について様々な研究が行われてきた。その中には、ソフトウェア信頼度成長曲線 [2] のような、ソフトウェア進化のある側面をソフトウェア開発履歴と統計モデルで分析する研究が存在する。このようなアプローチの利点は、適切な統計モデルを見つけてソフトウェア開発履歴に当てはめることができれば、統計モデルを決定する数個の母数で表現でき、ソフト

表 1. 本稿における略語

No.	略語	意味
1	Lognorm(μ, σ)	母数 μ, σ の対数正規分布
2	Zipf(α)	母数 α の Zipf 分布 [5][6]
3	Poisson(λ)	母数 λ のポアソン分布

ウェア開発履歴の構造に関してより深い解釈が可能になることである [3]。これは進化の進みや停止に対する深い分析につながり、OSS の製品への採否に資すると考えられる。

しかし、OSS の場合はソフトウェア進化を担う開発者の参加・離脱が自由であり、開発に伴う作業量も開発者間で大きく異なる点が OSS 開発の特徴として指摘されている [4]。これは、一般的にプロジェクトへの開発者の参加・離脱が上位者によって決定され、作業量も平準化が図られる企業等でのソフトウェア開発と異なっている。特に作業量のばらつきについては、OSS 開発プロジェクトの持続性の観点から重要性が指摘されている [4]。即ち、OSS 開発における作業量の分析は、OSS 進化を理解する上で重要である。そこで、本研究では、OSS 開発履歴に対して、適した統計モデルを選定してフィッティングし、その結果を通して OSS 進化を理解することを目的とする。

なお、本稿における略語を表 1 に示す。

2. 研究のアプローチ

データに適した統計モデルを選択する行為は、モデル選択 [7] と呼ばれる。モデル選択においては、データを説明

する複数の統計モデルを用意し、何らかの選択基準に則り、適した統計モデルを選ぶ。本稿では、OSS 進化という現象の理解を目的とするため、モデルの評価指標として、モデルのフィッティングの良さを評価する WBIC[8]を導入する。

加えて、OSS の進化を分析するためのデータとして、各コミットのコミット履歴を用いる。コミット履歴はソースコード変更の履歴であり、誰がいつ何を変更したかを記録したものである。

また、本研究では、競合 OSS 登場への対応や大小様々なバージョンアップなど、多様な進化を経験した PyMC^{*1}を解析する。

PyMC を対象に、コミット履歴を説明する統計モデルを複数用意し、WBIC を指標としたモデル選択により最もフィットしている統計モデルを抽出する。そして、抽出したモデルのフィッティング結果を分析することで、OSS 進化を理解する。

3. 本研究における要素技術

本項では、本研究で用いる要素技術である WBIC について述べるが、まず基礎となる BIC (Bayesian Information Criterion) [9] について述べる。BIC は、データに対する統計モデルと事前分布の組の適切さを評価する指標であり、以下の式 (1) により求められる。

$$\text{BIC} = -2\log\text{lik} + k\ln(n) \quad (1)$$

- loglik: データに対する統計モデルの対数尤度
- k: 統計モデルの母数の数
- n: データ数

以下が BIC を利用する上での注意事項である。

1. BIC 基準で統計モデルを評価する場合、BIC が小さいモデルが、モデルの複雑さを考慮した上でデータによくフィットしたモデルとなる。また、取得したデータを生成した理想的な「真の分布」が存在したとした場合、BIC 基準で統計モデルを評価すると「真の分布」と近いモデルを選ぶ性質を持つ。

2. 式 (1) はモデルの母数の数とデータサイズに依存する。そのため、同じモデルであっても、与えたデータの数異なれば BIC が異なる。
3. BIC でモデル選択をする際は、同じデータに対し異なる統計モデルを適用し、BIC の大小を比較する。
4. BIC を用いるためには、統計モデルが正則で、事後分布が正規分布で近似できる必要がある。

このような性質を持つ BIC を一般化したものが、Watanabe が提案した WBIC[8] である。WBIC の定義式を式 (2) に示す。注意事項の 1 から 3 は WBIC においても共通するが、WBIC に 4 のような制約は無い。OSS の開発履歴データが、注意事項の 4 番のような制約を守っている保証は無い。加えて、「真のモデル」に近いモデルを選ぶという、OSS 進化現象の理解に適した性質を WBIC は持っている。よって、WBIC を本研究にて用いる。

$$\text{WBIC} = \frac{\int nL_n(w) \prod_{i=1}^n p(X_i|w)^\beta \phi(w) dw}{\int \prod_{i=1}^n p(X_i|w)^\beta \phi(w) dw} \quad (2)$$

- n: データ数
- $\beta: 1/\log(n)$
- L_n : 対数損失関数
- $p(X|w): w$ を母数とする統計モデル
- $\phi(w)$: 事前分布

WBIC も、BIC と同じく WBIC が小さいモデルが高評価となる。

4. PyMC コミット履歴解析

4.1. PyMC 来歴

2 項に示したように、PyMC は競合 OSS への対応や大小様々なバージョンアップなど、様々な進化を経験した OSS である。そこで、以下に PyMC の簡単な来歴 [10] を示す。

- 2003 年：計算ベイズ統計用 OSS として開発開始
 - メトロポリス・ヘイスティングス法 [11][12] によるサンプリングアルゴリズムを採用
- 2006 年：GitHub 上での開発開始 (2.0 系)
- 2012 年：確率プログラミング言語 OSS・stan 登場

^{*1} <https://www.pymc.io/welcome.html> (2025-03-11)

– ハミルトニアンモンテカルロ (HMC:Hamiltonian Monte Carlo) 法の変法である NUTS (No-U-Turn Sampler) [13] がサンプリングアルゴリズムとして搭載された stan^{*2}が登場, PyMC 再設計開始

- 2017年：3.0版正式リリース
 - 再設計が完了し, NUTS 等を搭載した
- 2022年：4.0版, 5.0版正式リリース
 - 4.0版で開発終了した数値計算ライブラリである theano^{*3}を Aesara^{*4}に乗換
 - 5.0版で Aesara をフォークして PyTensor^{*5}として PyMC プロジェクトで維持することにし, 数値計算ライブラリを PyTensor に再乗換

本 OSS は 2025 年 3 月現在も進化が続いており, ガイドとなる書籍も発売されている等日本でも広く用いられている. 次節にて, 解析対象となる PyMC 開発履歴の基礎的な解析について示す.

4.2. PyMC コミット履歴の基礎的解析

まず, 以下の図 1 に PyMC のコミット履歴を示す. 本図は, 横軸に年, 縦軸にコミット回数を取り, 縦軸に PyMC の累積コミット回数を取ったものである. また, 前項にて紹介した PyMC における主要なイベントを図中に記入した.

次に, コミットを実際に行う存在である, コミッタによるコミット履歴を図 2 に示す. 本図は, 横軸に年, 縦軸にコミット回数を取り, PyMC に参画した全コミッタのコミット履歴を折れ線グラフで示したものである.

図 1 より, stan がリリースされた直後に累積コミット履歴の傾きが増加し, さらに PyMC3.0 版の α, β 版リリース直後に指数関数的にコミットが増加している. そして, 図 2 より, PyMC3.0 版がリリースされた 2017 年にコミット回数のピークを示したコミッタが数多く存在する. 対して, 2018 年以降, 累積コミット履歴の傾きは減少し, コミット回数のピークを示したコミッタの数も減少して

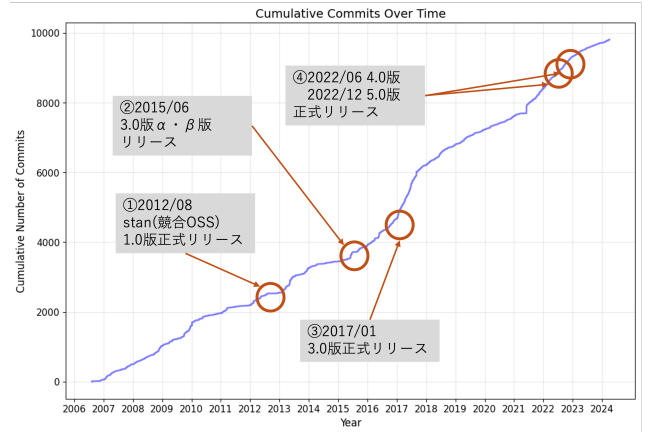


図 1. PyMC の累積コミット履歴

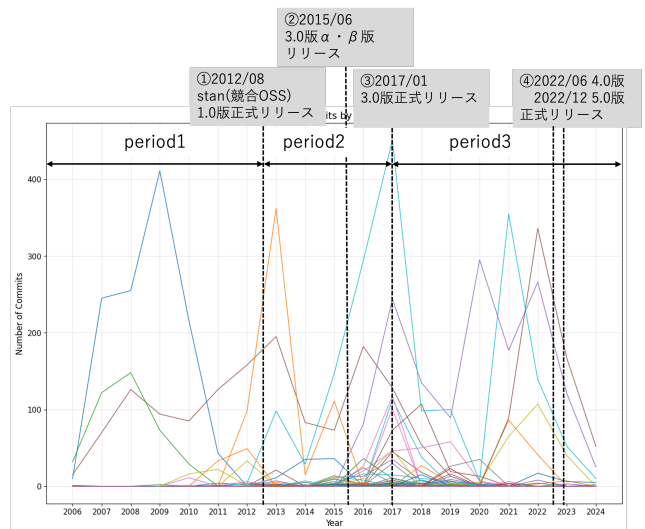


図 2. PyMC のコミット履歴 (コミッタ単位で集計)

いる.

以上より, PyMC の進化は, 以下の 3 つの期間に分かれると推測する. WBIC を統計モデルとデータに適用できる条件の 1 つに, データが独立同分布であることが挙げられる. 性質が異なるデータを混ぜると, 独立同分布が崩れる. そのため, 以降データを各期間ごとに 3 分割して解析を行う.

- period1:
 - ~2012/08/30
 - 競合 OSS である stan が現れるまでの期間
- period2:

^{*2} <https://mc-stan.org/> (2025-03-11)

^{*3} <https://github.com/Theano/Theano> (2025-03-11)

^{*4} <https://github.com/aesara-devs/aesara> (2025-03-11)

^{*5} <https://github.com/pymc-devs/pytensor> (2025-03-11)

- 2012/08/31～2017/01/09
- ソフトウェアを再設計し、3.0 版を開発する期間
- period3:
 - 2017/01/10～
 - 3.0 版リリース以降の期間

各 period において、コミッタごとの活動のばらつきの程度を把握するため、各コミッタの累積コミット件数を図 3～図 5 にグラフ化した。各グラフは横軸がコミット回数の順位であり、縦軸はコミット回数であり、両対数をとっている。

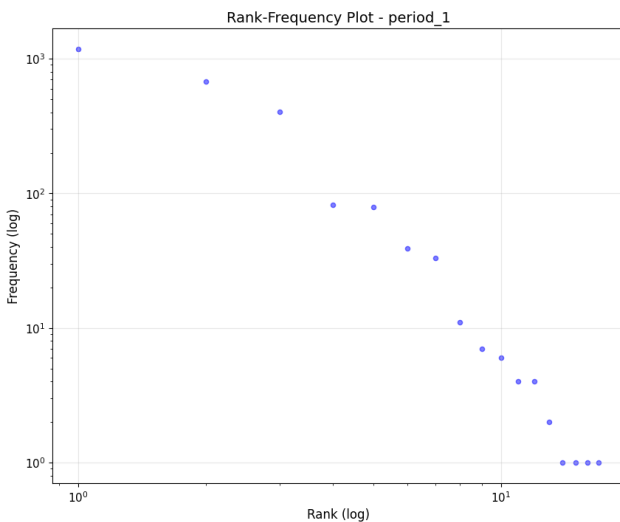


図 3. period1 における各コミッタのコミット回数順位対コミット回数グラフ（両対数）

各図より得られた知見を以下に示す。

- 各 period において、他のコミッタより 10 倍～1000 倍オーダーのコミットをするコミッタがいる一方、1～数回のコミットで離脱するコミッタがいる。（図 3～図 5）
 - 一般的にサンプルの値が大きくばらつくときとされている、裾が重い分布をモデルに用いる必要がある。
- コミット回数上位のコミッタは、一度コミットが上昇傾向になると、ピークを迎えるまでそのまま上昇を続ける傾向にある。
 - コミッタ個人のコミットが増加する様を表現で

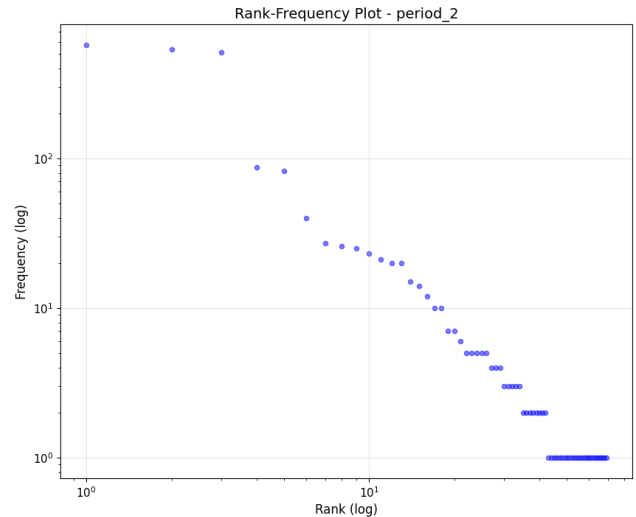


図 4. period2 における各コミッタのコミット回数順位対コミット回数グラフ（両対数）

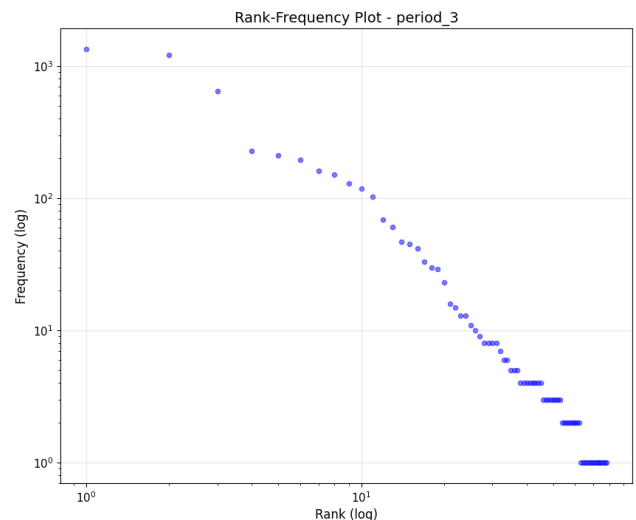


図 5. period3 における各コミッタのコミット回数順位対コミット回数グラフ（両対数）

きる分布をモデルに用いる必要がある。

- 図 4 と図 5 において、コミット回数順位が 2 位と 3 位のコミッタや低順位のコミッタらが各モデルの推定結果にフィットしていないが、これらはコミッタを層別して解析することで解決する可能性がある。

以上を踏まえ、次節にて使用する統計モデルを決定

する。

4.3. 使用するモデル概要

本研究で使用するのは、Zipf 分布 [5][6]、対数正規分布、ポアソン分布の 3 つである。Zipf 分布と対数正規分布は、一部のコミッタが他の 10~1000 倍オーダーのコミットをするようなデータを表現する統計モデルの代表例であり、これらの分布に至る過程を説明するモデルも提案されているものである。ポアソン分布は、コミット回数をコミッタ全員の平均値で表そうとした際に用いられる代表的なモデルである。

1. 対数正規分布モデル

- 以下の式で表されるモデルである。

$$Data \propto \text{Lognorm}(\mu, \sigma) \quad (3)$$

- 本モデルは、データが母数 μ, σ の対数正規分布に従うとしたものである。対数正規分布は裾が重い分布として一般的に知られている。
- もし本モデルがモデル選択で選ばれた場合、コミッタのコミット回数の成長がジブラ過程に従うと解釈できる。ジブラ過程は、都市の人口増加を説明したモデルである。もっとも基本的なジブラ過程モデルでは、あるタイムステップ $t+1$ における都市の人口 $x(t+1)$ は、 $x(t)$ と $x(t)$ に依存しない係数 $b(t)$ に比例して増加する。即ち、 $x(t+1) = b(t)x(t)$ となる。都市をコミッタ、人口をコミット回数に置き換えると、OSS 開発におけるコミット回数の増加を模擬したモデルと解釈できる。

2. Zipf 分布

- 以下の式で表されるモデルである。

$$Data \propto \text{Zipf}(\alpha) \quad (4)$$

- 本モデルは、コミット回数が冪分布の一種である、母数 α の Zipf 分布に従うとしたものである。冪分布は裾が重い分布として一般的に知られている。
- もし本モデルがモデル選択で選ばれた場合、コミッタのコミット回数の成長が Simon モデル [14] に従うと解釈できる。もっとも基本的な

Simon モデルは、都市の人口増加を説明したモデルである。確率 π である都市の人口が増加し、確率 $1 - \pi$ で都市が一つ増加する。人口の増加において、どの都市の人口が増えるかは、各都市の人口に比例した確率で決定される。都市をコミッタ、人口をコミット回数に置き換えると、OSS 開発におけるコミット回数の増加とコミッタの参入を模擬したモデルと解釈できる。

- もしくは、ジブラ過程において、ある一定確率 q でコミットの増加がリセットされるなど、ジブラ過程に変形を加えたモデル [15] と解釈できる。

3. ポアソン分布モデル

- 以下の式で表されるモデルである。

$$Data \propto \text{Poisson}(\mu) \quad (5)$$

- 本モデルは、データが母数 λ のポアソン分布に従うとしたものである。
- もし本モデルがモデル選択で選ばれた場合、コミッタの平均コミット回数が全員同じであり、コミット回数期待値の視点では全員均一であると解釈できる。

4.4. 解析結果

前節のモデルをコミット履歴データに適用し、NUTS により母数推定を行った。NUTS を用いたのは以下の理由による。

- 3 項で述べたように、WBIC は積分を含んでいる。これは最尤法のような点推定とは相性が悪く、サンプルを大量に生成する NUTS の利用が適している。
- NUTS は乱数を利用した手法であるが、高次積分における次元の呪いに強く、一般的なモンテカルロ法よりも積分に強い。

解析結果を以下の表 2 に示す。全 Period において、Zipf 分布の WBIC が最小となった。

また、母数推定を行なった各モデルをコミットデータにプロットした結果を図 6~図 8 に示す。

図表よりわかることを以下に示す。

- 全 period において、Zipf 分布の WBIC が一番小さ

表 2. WBIC の比較

No.	モデル	WBIC		
		Period1	Period2	Period3
1	対数正規分布	14.39	14.16	17.89
2	Zipf 分布	13.39	12.81	16.97
3	ポアソン分布	596.06	276.55	541.62

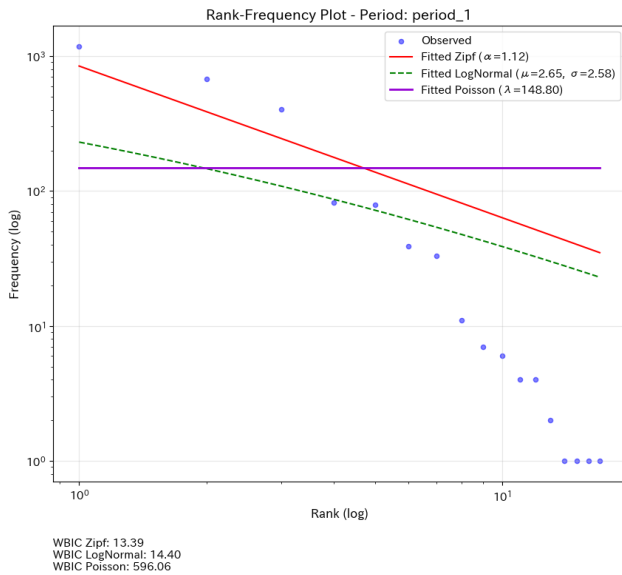


図 6. period1 における各モデルフィッティング結果

く、最もコミット履歴に適合する。これは、今回使用した統計モデルの中では Zipf 分布が「真のモデル」に最も近いことを意味する。(表 2)

- period1 は zipf 分布、対数正規分布ともにフィットしていないが、これはデータ数が少ないことと、コミット回数上位 3 名に引っ張られたためと推測できる。(図 6)
- ポアソン分布は全 period にてデータにフィットしていない。即ち、各 period におけるコミッタのコミット回数期待値が均一であると見做せない。(図 6～図 8)
- 全 period において Zipf 分布が対数正規分布よりもフィットしていたが、これはコミッタのコミット回数が上昇傾向にある時は、コミット回数が乗算的ではなく加算的に増加することを意味する。(図 6～図

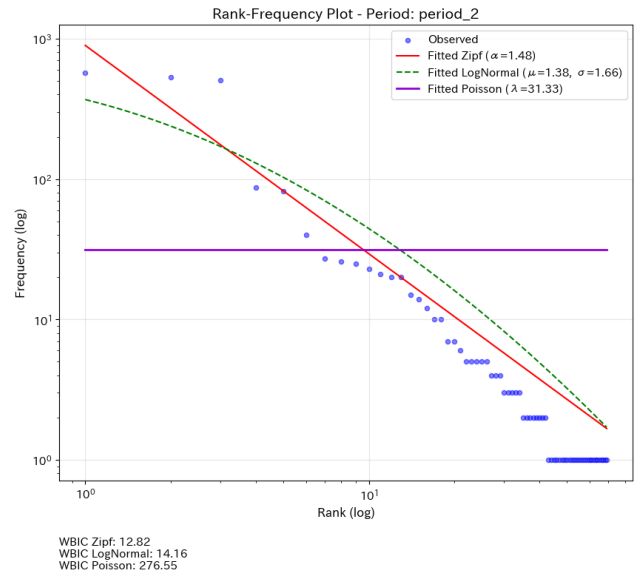


図 7. period2 における各モデルフィッティング結果

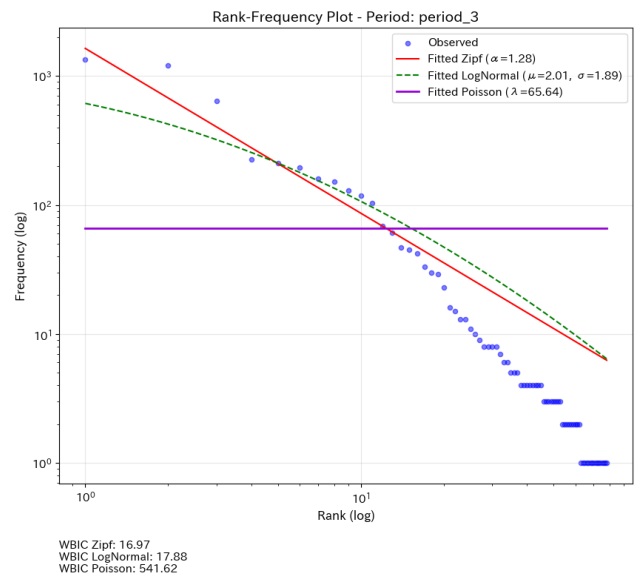


図 8. period3 における各モデルフィッティング結果

8)

5. 考察

- 前項より、本稿でデータに適用した統計モデルの中で、Zipf 分布が最もコミット履歴に適合する結果を

得た。よって、PyMC のコミット履歴は、Simon モデルや改良されたジブラ過程モデルから生成された可能性があるとして解釈することができる。

- Zipf 分布モデルの母数を前項で推定した結果、どの period でも母数 α が 2 を下回った。Zipf 分布は、以下に示す通り、母数 α が 2 未満のとき期待値が、3 未満の時は分散が存在しない。
 - Zipf 分布の確率質量関数を示した式 (6) より、Zipf 分布の確率質量関数はリーマンゼータ関数を発散させないために母数が 1 以上でなければならない。
 - Zipf 分布の期待値を示した式 (7) より、期待値が存在するためには母数 α が 2 以上でなければならない。
 - Zipf 分布の分散を示した式 (8) より分散が存在するためには母数 α が 3 以上でなければならない。

本稿の結果は、どの period でも期待値と分散が存在しなかった。これは、PyMC におけるコミット回数の平均や分散を計算し、PyMC のコミット回数を代表する値として用いることができないことを意味する。

$$P(k; \alpha) = \frac{1/k^\alpha}{\sum_{n=1}^{\infty} 1/n^\alpha} = \frac{1/k^\alpha}{\zeta(\alpha)} \quad (6)$$

$$E[X] = \sum_{k=1}^{\infty} k \cdot P(k; \alpha) = \frac{\sum_{k=1}^{\infty} 1/k^{\alpha-1}}{\sum_{n=1}^{\infty} 1/n^\alpha} = \frac{\zeta(\alpha-1)}{\zeta(\alpha)} \quad (7)$$

$$V[X] = E[k^2] - (E[k])^2 = \frac{\zeta(\alpha-2)}{\zeta(\alpha)} - \left(\frac{\zeta(\alpha-1)}{\zeta(\alpha)} \right)^2 \quad (8)$$

- $P(k; \alpha)$: 確率質量関数
- $E[X]$: Zipf 分布に従う確率変数 X の平均 (期待値)
- $V[X]$: Zipf 分布に従う確率変数 X の分散
- k : 順位 (1, 2, 3, ...)
- α : 分布の母数
- $\zeta(\alpha)$: リーマンゼータ関数
- コミット履歴に最も適合する統計モデルが Zipf 分布であることは、PyMC のコミット回数の多くをトップ層が占めていることを意味している。Zipf 分布の

母数 α は、コミット回数のトップ層への集中度合いと解釈することができ、大きいほどトップ層にコミット回数集中していることになる。これは、以下のように応用可能である。

- 企業などで開発しているソフトウェアでコミット回数が Zipf 分布に従い、 α が大きい場合、一部の開発者に作業が集中していると評価することができる。これを用いて、プロジェクトマネージャが作業量を平準化する等の施策を打つことができる。
- 妥当性についての考察を以下に示す。
 - 内部妥当性：
 - * 本研究は、取得したコミット履歴を全て解析に用いた。よって、サンプルサイズやサンプルの偏りに起因する内部妥当性への脅威は存在しない。
 - 外部妥当性：
 - * 本研究は、PyMC のみを対象に解析を行ったため、結果を全 OSS に敷衍できるとは限らない。しかし、統計モデルを立てて WBIC で評価する手法自体は汎用性がある。
 - * 本研究では 3 つの period にデータを分割した。分割はデータが独立同分布になるよう行っており、この方針は汎用性がある。

6. 関連研究

OSS 開発者の貢献量に対して冪分布を当てはめた研究として増田らの研究 [16] が挙げられる。増田らの研究において、COCOMO を用いて OSS 開発コストを推定する一環で冪分布を用い、また集計対象としてコミット回数とソースコードライン長から算出した貢献量を用いている。そのため単純な援用はできないが、50 の OSS において貢献量が冪分布を示したため、コミット回数も様々な OSS で冪分布に従う可能性がある。

Mockus ら [17] は、Apache 開発プロジェクトにおける開発履歴を調査した。その結果、全ソースコードの追加のうち、88% と全バグ修正のうち 66% が全体の 4% の開発者 (15 人/388 人) によって行われたことが明らかになった。このように、少数の開発者によって多くの作業

が行われるのは冪分布の性質で説明でき、本稿の成果とも整合性がある。

Koch ら [18] は、様々な OSS の進化について分析し、進化が super-linear な性質を示す場合、プロジェクトとソースコードの規模の他にプロジェクト内での作業量の不均衡に正の相関があることを明らかにした。この研究の結果も、本稿の結果と整合性がある。

これらの研究を総括すると、様々な尺度で計測した OSS 開発における貢献や作業の量は、少数の開発者によってなされている。本研究では、それを複数の統計モデルで表現し、さらに WBIC を用いて最適なモデルを選択した点に新規性がある。

7. 結論

本稿では、PyMC のコミット履歴データに対して、WBIC を評価指標としたモデル選択を行うことで、Zipf 分布・対数正規分布・ポアソン分布の中で Zipf 分布がもっとも合うことを明らかにした。これは、PyMC のコミット履歴が Simon モデル、もしくは改良されたジブラ過程モデルにより生じた可能性があると解釈できる。

8. 今後の課題

今後、より詳細な分析を PyMC コミット履歴データに対して行う必要がある。なぜならば、第一に、Simon モデルや改良されたジブラ過程など、どのようなメカニズムにより当該データが生じたのか明らかにする必要があるためである。第二に、本研究ではコミットの数で OSS 進化を分析したが、コミットの内容も重要であり、コミット各々の作業内容も踏まえた分析を行う必要があるためである。

4 節に示したように、図 4 と図 5 における各モデルの推定結果に一部データがフィットしていないが、これはコミットの層別で解決する可能性がある。そこで、作業の内容などの詳細なデータを分析することでコミットを層別してさらなる解析を行う。

PyMC だけではなく、様々な OSS について本稿の解析を行い、本稿の結果の一般性を確認する必要がある。

参考文献

- [1] 片山卓也. 発展ドメイン: ソフトウェア発展のための理論的枠組み. コンピュータソフトウェア, Vol. 21, No. 3, pp. 11–21, 2004.
- [2] A. L. Goel and K. Okumoto. Time-dependent error-detection rate model for software reliability and other performance measures. *IEEE Transactions on Reliability*, Vol. R-28, No. 3, pp. 206–211, 1979.
- [3] 玉井哲雄, 中谷多哉子. ソフトウェア進化プロセスの統計モデル. コンピュータソフトウェア, Vol. 21, No. 3, pp. 159–168, 2004.
- [4] 伊原彰紀, 大平雅雄. 『オープンソースソフトウェア工学』シリーズ オープンソースソフトウェア工学. コンピュータソフトウェア, Vol. 33, No. 1, pp. 28–40, 2016.
- [5] George K. Zipf. *Human Behaviour and the Principle of Least Effort*. Addison-Wesley, 1949.
- [6] 小野寺夏生. Zipf-bradford 分布 — 計量情報学における基本的モデル. オペレーションズ・リサーチ, Vol. 31, No. 3, pp. 144–151, 1986.
- [7] 松嶋敏泰. 統計モデル選択の概要. オペレーションズ・リサーチ, Vol. 41, No. 7, pp. 369–374, 1996.
- [8] Sumio Watanabe. A widely applicable bayesian information criterion. *Journal of Machine Learning Research*, Vol. 14, No. 27, pp. 867–897, 2013.
- [9] Gideon Schwarz. Estimating the Dimension of a Model. *The Annals of Statistics*, Vol. 6, No. 2, pp. 461 – 464, 1978.
- [10] pymc-devs. pymc-devs/pymc リポジトリ. <https://github.com/pymc-devs/pymc>. 2025-03-11 閲覧.
- [11] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, Vol. 21, No. 6, pp. 1087–1092, 1953.
- [12] W.K. Hastings. Monte carlo sampling meth-

- ods using markov chains and their applications. *Biometrika*, Vol. 57, No. 1, pp. 97–109, 1970.
- [13] M.D. Hoffman and A. Gelman. The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo. *Journal of Machine Learning Research*, Vol. 15, pp. 1593–1623, 2014.
- [14] HERBERT A. SIMON. On a class of skew distribution functions. *Biometrika*, Vol. 42, No. 3-4, pp. 425–440, 12 1955.
- [15] 富田真治. 確率モデルの変数制御性を利用した人口分布の再現. PhD thesis, 北陸先端科学技術大学院大学, 2008.
- [16] 増田礼子, 森本千佳子, 松尾谷徹, 津田和彦. 大規模オープンソース・ソフトウェアプロジェクトにおける開発効率の計測. 電気学会論文誌C (電子・情報・システム部門誌), Vol. 138, No. 8, pp. 1011–1019, 2018.
- [17] Audris Mockus, Roy T. Fielding, and James D. Herbsleb. Two case studies of open source software development: Apache and mozilla. *ACM Trans. Softw. Eng. Methodol.*, Vol. 11, No. 3, p. 309–346, July 2002.
- [18] Stefan Koch. Software evolution in open source projects—a large-scale investigation. *Journal of Software Maintenance and Evolution: Research and Practice*, Vol. 19, No. 6, pp. 361–382, 2007.