オープンソース開発におけるソースコード変更予測に向けた成長曲線モデルの多段的利用について

山下彰子[†1] 阿萬裕久[†2]

本稿では成長曲線を多段的に活用することで，複雑に進化していくオープンソースソフトウェアに対し，その開発状況のモデル化を試みている．

On Multistage Model of Growth Curves toward Source Code Change Prediction in Open Source Development

Akiko Yamashita[†1] and Hirohisa Aman[†2]

This paper proposes a multistage model of growth curves for modeling complex status of open source software development.

1. はじめに

ソフトウェア品質マネジメントを実践する上で開発状況や品質の定量的な把握は常に必要である．実際，信頼性という視点では，テストで検出された障害件数の累積値を入力としたさまざまな信頼度成長モデル[1)]が研究され，開発現場で広く使われている．例えば，経験的なものとしてはゴンペルツ曲線やワイブル曲線といったものがあり，より理論的なものとしては障害検出を非同次ポアソン過程でモデル化した遅延 S 字形モデルや習熟 S 字形モデル等がある．

近年そういった成長曲線モデルをオープンソース開発にも適用する研究も行われてきている．オープンソース開発の場合，不特定多数のユーザからの障害報告や改善・拡張要求を受けて開発が進んでいく．そのため，特定の企業や組織の中に閉じた開発に比べて開発の形態が複雑化しており，それだけ開発状況や品質の把握が難しくなっている．それでもなお，今日のソフトウェア業界に対する多様なニーズもあり，産業界でもオープンソースソフトウェアを製品やサービスの一部に活用する場面も珍しくない．つまり，オープンソースソフトウェアの開発状況や品質の定量化には一定のニーズがあると考えられる．

我々はこれまで，オープンソースソフトウェアに対する障害報告件数やソースファイルの変更回数等を成長曲線を使ってモデル化するという研究を行ってきた[2)]．しかし，多くのデータを分析していく中で，単一の成長曲線では（一つのソフトウェアにおける）データ全体をうまく説明できないという場面に遭遇することも少なくなかった．そこで本稿では，そういったデータに対し，オープンソース開発の特色も考慮した上で複数の成長曲線を多段的に活用することを提案する．

2. 成長曲線の多段的な利用

本稿ではリポジトリへのコミット回数[⋆1]をモデル化対象の実データとして取り扱う．実際のオープンソース開発プロジェクトからは他にもさまざまなデータを収集可能であるが，ここでは紙面の都合上，コミット回数に限定して議論する．

図 1 はオープンソースソフトウェア Nagios[⋆2] のリポジトリに対する月毎の累積コミット回数とそれに当てはめたゴンペルツ曲線を示している．図 1 から明らかなように，開発の初期から現在に至るまでのコミット回数の推移（成長）は同図のゴンペルツ曲線ではうまく説明できない．ゴンペルツ曲線の場合，大まかには “初期（立ち上がり），中期（成長），後期（収束）”

†1 愛媛大学工学部情報工学科
Department of Computer Science, Faculty of Engineering, Ehime University
†2 愛媛大学総合情報メディアセンター
Center for Information Technology, Ehime University

⋆1 リポジトリとしては Git を対象としている．ただし，一度のコミットで 2 個のファイルが更新されている場合は 2 回と数える．
⋆2 http://www.nagios.org/. 情報基盤システムの監視を行うアプリケーションソフトウェアであり，執筆時点で開発・保守は 10 年以上続いている．

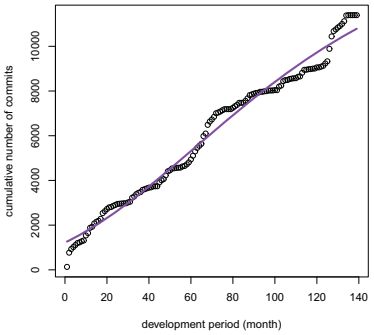

図 1　Nagios の累積コミット回数とゴンペルツ曲線

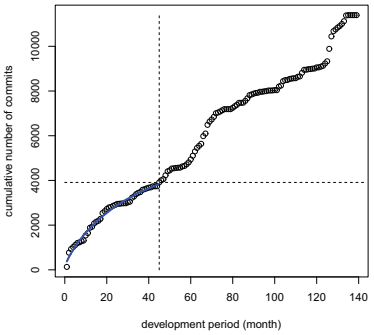

図 2　最初の 45 ヶ月目までワイブル曲線を当てはめた例

の三ステージに分かれるが，曲線が当てはまらない限り，そのような時期を当てはめて考察することもできない．しかし，図 1 のデータでは 40 ヶ月目付近や 100 ヶ月目付近で一時的な収束状態にあるようにも見える．つまり，ある程度の期間でいったんは開発が概ね収束し，その後再び開発が活発化するというサイクルが繰り返されているようにも見える．そこで本稿では，開発期間全体に対して一つの成長曲線に当てはめて考えるのではなく，途中で一時的な収束が見られた時点で “仕切り直し” を行うことを提案する：

(1)　$s \leftarrow 1$

(2)　開発開始 s ヶ月目を改めて 1 ヶ月目と見なし，そこから x ヶ月後までのデータに対して成長曲線を当てはめる．x は 12 以上で $s+x$ が最終月となるまで全パターンを試す．

(3)　(2) の中で実データとの平均誤差（次式で定義）が最小となる成長曲線と x の組を見つける：

$$\frac{1}{x+1}\sum_{i=s}^{s+x}(y_i-\widehat{y}_i)^2 \ ,$$

ただし，y_i は i ヶ月目での累積コミット回数であり，$\widehat{y}_i$ は成長曲線を使った推定値である．

(4)　$s+x$ が最終月でなければ $s \leftarrow s+x+1$ として (2) へ戻る．

□

今回はロジスティック曲線，ゴンペルツ曲線，ワイブル曲線，指数形成長モデル，遅延 S 字形成長モデル及び習熟 S 字形成長モデルの 6 種類の成長曲線を使った適用実験を行った．

まず，1 ヶ月目のデータを基点として，そこから順に注目期間を延ばしつつ，全種類の成長曲線を当てはめるという実験を行ったところ，最初の 45 ヶ月目までのデータにワイブル曲線を当てはめるのが最も誤差が小さいことが分かった．そこでその月でいったん切り替えを行い，46 ヶ月を改めて 1 ヶ月目と見なして（**図 2**）再び曲線の当てはめを行っていった．

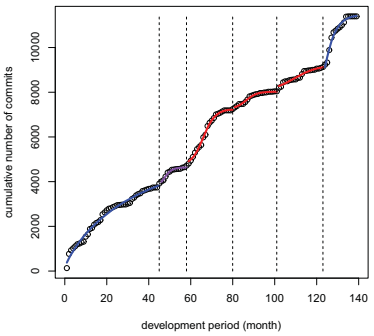

図 3　順に成長曲線を当てはめた結果

結果として**図 3** のように 6 ステージに分けて当てはめを行うのが誤差が最小となることが分かった．

3. おわりに

一つの成長曲線を開発期間全体にわたって使用するのではなく，状況に応じてモデルを切り替えていく方がより現実的ではないかと考えられる．ワークショップではこの種の解析の有効性について議論を深めたい．

参考文献

1) 山田茂：ソフトウェア信頼性モデル—基礎と応用，日科技連 (1994).

2) Aman, H.: A Proposal of NHPP-Based Method for Predicting Code Change in Open Source Development, *Proc. Joint Conf. 21st Int'l Workshop on Software Measurement and 6th Int'l Conf.on Software Process and Product Measurement (IWSM-MENSURA 2011)*, pp. 38–47 (2011).