The 32nd Annual Conference of the Japanese Society for Artificial Intelligence, 2018

ハイブリッド制約言語 HydLa における 非線形常微分方程式の表現とその記号付き精度保証計算

Validated Simulation of Parametric Nonlinear ODE on a Hybrid Constraint Language HydLa

増田健太[*1]
Kenta MASUDA

上田和紀[*2]
Kazunori UEDA

[*1] 早稲田大学大学院基幹理工学研究科情報理工・情報通信専攻
Graduate School of Fundamental Science and Engineering, Waseda University

[*2] 早稲田大学理工学術院情報理工学科
Faculty of Science and Engineering, Waseda University

The purpose of this research is to obtain an over-approximated reachability set of a nonlinear ODE while conserving symbolic parameters in the approximation. Conserved symbolic parameters represent the behavior of the solution, which is useful for analyzing the characteristics of nonlinear phenomena influenced by parameters. In this research, first we generate parallelotope over-approximation of nonlinear functions composing the nonlinear ODE, and using the parallelotopes, construct a hybrid system as its linear over-approximation. Then we analyze the generated hybrid system symbolically, thus we obtained a guaranteed-accuracy solution of the parametric nonlinear ODE.

1. はじめに

非線形な常微分方程式には一般に解析解が存在せず，非線形現象においてパラメータの変化が解にどのような影響を与えるのかを調べるのは容易ではない．

本研究は，非線形な常微分方程式を過大近似された HydLa[1]モデルに変換し，これについて記号的な解析を行うことでパラメータを記号として残した精度保証解を計算できるようにすることを目的とする．

HydLa とはハイブリッドシステムのシミュレーションや検証を行うための制約に基づくモデリング言語である．ハイブリッドシステムは，連続的な運動と離散的な状態遷移を合わせ持つ動的システムのことで，サイバーフィジカルシステム（物理環境との相互作用を持つ制御システム）などのモデル化に用いられる．

HydLa で書かれたモデルを実行する処理系として，HyLaGI[2]という C++による実装が存在する．これは Mathematica を用いた記号的解析を行うという特徴を持っている．HyLaGI はその特徴から，ハイブリッドシステムの厳密な解軌道を求めることができ，また初期値にパラメータを含むようなモデルについてもそのパラメータを記号として残せるという利点を持つ．

提案手法では非線形な常微分方程式について，まずこれを構成する非線形関数を複数の平行体により包含し，これをもとに幅を持つ線形な常微分方程式を複数つなげたハイブリッドシステムを構築する．そして，このハイブリッドシステムについて HyLaGI による記号的解析を行うことで，パラメータを記号として残した精度保証解を計算する．また本研究では評価実験として，単振り子の運動の解析を行った．実験では振り子の長さをパラメータとして，実際に記号を部分的に残した精度保証解が求められることを確認した．

連絡先：増田健太，早稲田大学大学院基幹理工学研究科情報理工・情報通信専攻，〒 169-8555 新宿区大久保 3-4-1 63 号館 5 階 02 号，03-5286-3340，kenta(at)ueda.info.waseda.ac.jp

2. HyLaGI における精度保証計算

HyLaGI には離散変化時刻が記号的に求められないモデルについて，記号計算と精度保証計算を用いてパラメータを含むシミュレーションを行う区間モード[3]が実装されている．以下に区間モードで用いられている精度保証計算手法について説明をする．

2.1 区間モード

HyLaGIの区間モードは，常微分方程式が線形なハイブリッドシステムについて，区間ニュートン法[4]とアフィン演算[5]を用いたシミュレーションを行うことで，パラメータ間の一次の依存関係を保存した精度保証解を計算するものである．これにより，離散変化時刻が記号的に求められないモデルであってもパラメータを含むシミュレーションが可能となる．

2.2 区間ニュートン法

区間ニュートン法とは，ニュートン法を区間演算により拡張した手法であり，代数方程式の精度保証解を求めることができる．区間モードで実際に使われている拡張区間ニュートン法という手法は，非線形関数のある定義域に含まれる根を確実に求めることができるという性質を持っている．

2.3 アフィン演算

アフィン演算とは，区間演算では避けられない Wrapping Effect や Dependency Problem による精度の悪化を改善するために，値を単一の区間値で包含するのではなくアフィン式と呼ばれる一次の多変数多項式で包含して計算を行う手法である．

3. 提案手法

提案手法は大きく分けて，非線形関数の包含，ハイブリッドシステムへの変換，HyLaGI による精度保証計算の三つのステップからなる．この中で，非線形関数の包含はさらに 3.1-3.3 の三段階に分かれる．

以下にそれぞれについて説明をする. また, 非線形関数の平行体による包含を求める$ParallelotopeOverApproximation$と$GetParallelotope$のアルゴリズムをそれぞれ図1と図2に示す.

3.1 平行体を用いた非線形関数の包含

まず, 常微分方程式を構成する非線形関数について, アフィン演算を用いて評価することにより, これを包含する平行体を得る. 計算は, 次のような平均値形式を用いることで行う.

$$f_m(\boldsymbol{x}) \coloneqq f(x_m) + f'(\boldsymbol{x})(\boldsymbol{x} - x_m) \qquad (x_m \in \boldsymbol{x})$$

3.2 平行体の最適化

前節の計算によって求められた平行体は, 幅が十分に狭いものではない. したがって, この平行体の幅を縮める最適化を行う.

まず, 非線形関数上の定義域中のある一点を選び, この点を通り幅が 0 であるような平行体を考える. 次にこの幅 0 の平行体について, 元の平行体の幅を最大値として, 非線形関数を包含できるような最小の幅を二分探索によって求める. ここで, 平行体が非線形関数を包含しているかどうかは, 非線形関数が一変数関数ならば区間ニュートン法によって確かめることができる.

3.3 多分木による空間分割

提案手法では, 非線形関数を複数の平行体により包含する. これについて後の手順で隣接判定を行いやすいように, 与えられた定義域について多分木による空間分割を行う. ここでいう多分木とは, 空間の次元数Nについて, 分岐の数が2^Nとなるような木を指すものとする.

3.4 ハイブリッドシステムへの変換

前節までで得られた平行体を用いて, ハイブリッドシステムを構築する. まずそれぞれの平行体について, 常微分方程式の非線形関数をこれにより置き換えた線形常微分方程式を作る. そして, 現在のシステムの状態がどの平行体に属しているかを表す離散変数を導入し, この値をガード条件に取ることで微分方程式の場合分けを行う. 最後に, この離散変数をガード条件に取り, 連続変数が平行体の端に到達した瞬間に, 隣接する平行体に離散変数を移動するような離散変化を追加し, これをハイブリッドシステムの定義とする.

したがって, これを作るには平行体がどの平行体と隣接しているかを列挙する必要がある. これには, 前節で構築した多分木の構造的な情報を利用して隣接判定を行う.

3.5 HyLaGI による精度保証計算

以上の方法で構築したハイブリッドシステムは, HyLaGI により記号計算もしくは区間モードを用いて精度保証計算を行うことが可能である.

4. 評価実験

評価実験として, 単振り子の運動の解析を行った.

実験は, 初期値にパラメータを含まないケースと含むケースのそれぞれについて, 非線形関数を包含する平行体の個数を 3 個から 73 個まで変えながら行った. 重力加速度は$9.8[m/s^2]$とした. 以下にそれぞれの結果について述べる.

4.1 パラメータを含まないモデル

振り子の初期角度θ_0が20[deg]の場合と60[deg]の場合について, 紐の長さを$5[m]$として実験を行った. この結果について, それぞれの場合で単振り子の周期の計算を行った結果を次の

Input: 変数の定義域$[x_{min}, x_{max}]$, 平行体の個数num
Output: 平行体の集合$approx$

$mid \coloneqq (x_{min} + x_{max})/2$
$approx.push(GetParallelotope(x_{min}, mid))$
$approx.push(GetParallelotope(mid, x_{max}))$
while $approx.size < num$ **do**
  $elem \coloneqq PopLargest(approx)$
  $mid \coloneqq (elem.x_{min} + elem.x_{max})/2$
  $approx.push(GetParallelotope(elem.x_{min}, mid))$
  $approx.push(GetParallelotope(mid, elem.x_{max}))$
end while

図 1. $ParallelotopeOverApproximation$のアルゴリズム

Input: 変数の定義域$[x_{min}, x_{max}]$
Output: 平行体$parallelotope$

$x_{mid} \coloneqq (x_{min} + x_{max})/2$
$affine \coloneqq f_m([x_{min}, x_{max}])$
$y_{mid} \coloneqq f(x_{mid})$
$b_{mid} \coloneqq y_{mid} - affine.a \times x_{mid}$
$a \coloneqq affine.a$

$b_{min.lower} \coloneqq 0$
$b_{max.lower} \coloneqq affine.b.lower$
while $eps < |b_{min.lower} - b_{max.lower}|$ **do**
  $mid \coloneqq (b_{min.lower} + b_{max.lower})/2$
  if $Intersects(a, b_{mid} - mid, x_{min}, x_{max})$ **then**
    $b_{min.lower} = mid$
  else
    $b_{max.lower} = mid$
  end if
end while

$b_{min.upper} \coloneqq 0$
$b_{max.upper} \coloneqq affine.b.upper$
while $eps < |b_{min.upper} - b_{max.upper}|$ **do**
  $mid \coloneqq (b_{min.upper} + b_{max.upper})/2$
  if $Intersects(a, b_{mid} + mid, x_{min}, x_{max})$ **then**
    $b_{min.upper} = mid$
  else
    $b_{max.upper} = mid$
  end if
end while

$b \coloneqq [b_{mid} - b_{max.lower}, b_{mid} + b_{max.upper}]$
$parallelotope \coloneqq \{a, b\}$

図 2. $GetParallelotope$のアルゴリズム

表 1 と表 2 に示す. なお, 分割数は非線形関数の包含に使用した平行体の個数を表す.

結果を見ると, 分割数を上げるほど周期の区間幅が狭まっていることがわかる. ただし, 分割数を増やした時の区間幅の縮まり方については条件により差があった. この原因としては, 非線形関数を包含する平行体の幅がそれぞれ異なることから, 必ずしも分割数によって連続的に精度が良くなっていくわけではないということが考えられる.

分割数	振り子の周期	周期の区間幅
3	[4.521901060, 4.538531382]	1.66e−2
13	[4.522430438, 4.522560454]	1.30e−4
23	[4.522429294, 4.522448685]	1.94e−5
43	[4.522421323, 4.522427793]	6.47e−6
73	[4.522417420, 4.522417421]	1.46e−12

表 1. 分割数と周期の関係($\theta_0 = 20$[deg])

分割数	振り子の周期	周期の区間幅
3	—	—
13	[4.816213694, 4.818033176]	1.82e−3
23	[4.816477590, 4.816844693]	3.67e−4
43	[4.816416753, 4.816563980]	1.47e−4
73	[4.816417615, 4.816497212]	7.96e−5

表 2. 分割数と周期の関係($\theta_0 = 60$[deg])

4.2 パラメータを含むモデル

振り子の初期角度θ_0が20[deg]の場合と60[deg]の場合について実験を行った. また, 振り子の長さlにはパラメータとして$5 \le l \le 5.01$と$5 \le l \le 5.1$の範囲を設定し, そのそれぞれについてシミュレーションを行った.

この結果のうち一例として, 初期角度60[deg], 分割数 13, $5 \le l \le 5.01$の条件で実験を行った結果から得られた振り子の周期の式を次に示す. なお, 式に出てくる小数は見易さのために近似を行った結果であり, 内部表現は分数の形なので浮動小数点計算による計算誤差は発生していない.

$$T \in [4.78495, 4.85332] \ + \ 4.82725^{-4}\, p[l, 0, 1]$$

式に出てくる$p[l, 0, 1]$は振り子の長さを表すパラメータで, $[-1, +1]$の範囲の値を取り, $5 \le l \le 5.01$の範囲の値を表現する.

式を見ると, 振り子の周期が長さに関する一次式の形で得られていることがわかる. 式の一項目は単一の区間値であり, この区間幅が平行体の包含と実行時の精度保証計算により生じた誤差の合計を表している. そして, 二項目は$p[l, 0, 1]$の幅が 2 であることがわかっているので, このパラメータの係数を 2 倍した値がパラメータが解に与える変化量の大きさ, つまり, パラメータにより制御が可能な範囲の大きさを表している. したがって, 誤差に対するこの値の割合が大きいほど, パラメータの記号的性質を良く表現できていると考えられる.

以上の理由から, 実験結果として$5 \le l \le 5.01$のそれぞれのケースについて, 周期の式全体を評価した区間幅のうち, パラメータ項のみを評価した区間幅の大きさの割合を計算した. これを次の表 3 に示す. なお, $5 \le l \le 5.1$のケースについても実験を行ったが, その全ての実験において 1 周期まで計算を行う前に計算が失敗してしまったため結果は省略する.

表を見ると, 結果は予想に反して分割数が 13 の時が最も良く, 分割数を増やすとかえって途中で計算が失敗してしまうケースがあることがわかった. この原因としては, 分割数次第で関数の形が変わることや, 関数の形によって解を包含するアフィン式の形が変わることなど, 必ずしも精度とは関係の無い要素が解の唯一性証明の成否に与える影響が大きいということが考えられる.

また, 最も良い結果であった 13 分割のケースにおいても, 全区間幅に対するパラメータによる区間幅の大きさの割合はわずか 10%にとどまり, 誤差に対してパラメータが非常に小さな変化量しか表現できていないという結果となった.

分割数	$\theta_0 = 20$[deg] 全区間幅に対するパラメータの割合[%]	$\theta_0 = 60$[deg] 全区間幅に対するパラメータの割合[%]
3	6.17	—
13	10.3	2.55
23	3.81	1.40
43	1.32	—
73	—	—

表 3. パラメータが解に与える影響の大きさの比較

5. まとめと今後の課題

本研究では非線形関数が一変数関数からなる常微分方程式を, 幅を持った線形常微分方程式からなるハイブリッドシステムに変換し, 初期値のパラメータを部分的に残した精度保証解を計算することに成功した. しかし, 初期値にパラメータを含まないケースでは予想に近い結果が得られたものの, 初期値にパラメータを含むケースについては課題が多く残る結果となった。今後の課題としては以下に二つ示す.

5.1 多変数関数への対応

現在のアルゴリズムは一変数関数に適用することを前提としたものであるが, 平行体の最適化以外については多変数関数にも拡張することが可能だと考えられる.

5.2 初期値に大きな幅を持つパラメータへの対応

現在の計算が失敗する主な原因は解の唯一性保証に失敗するというものである. これは, 区間ニュートン法の結果が唯一の真の解を包含しているかを検証するという処理であるが, 関数の区間幅が広いほど失敗しやすくなるため, 初期値に大きな幅を持つパラメータに対応できないという問題が存在する. この問題に対して, 解の唯一性保証に失敗したらパラメータの区間を分割して再度計算を行うことで解決することが可能だと考えられる.

参考文献

[1] 上田和紀, 石井大輔, 細部博史: ハイブリッド制約言語 HydLa の宣言的意味論, コンピュータソフトウェア, Vol.28 No.1, 2011, pp.306-311.

[2] 松本翔太: Validated Simulation of Parametric Hybrid Systems Based on Constraints, 博士学位論文, 早稲田大学, 2017.

[3] 松本翔太, 上田和紀: Symbolic Simulation of Parametrized Hybrid Systems with Affine Arithmetic, 23rd International Symposium on Temporal Representation and Reasoning, 2016.

[4] Ramon E.Moore, R.Baker Kearfott, Michael J.Cloud: Introduction to Interval Analysis, Society for Industrial and Applied Mathematics, Philadelphia, 2009.

[5] de Figueiredo, L. H. and Stolfi, J. : Affine Arithmetic: Concepts and Applications, Numerical Algorithms, Vol. 37, Issue 1-4, 2004, pp. 147–158.