一般解説

非線形 Receding Horizon 制御の計算方法について

大塚敏之*

*大阪大学 大学院工学研究科 電子制御機械工学専攻
大阪府吹田市山田丘 2-1
*Department of Computer-Controlled Mechanical Systems, Graduate School of Engineering, Osaka University, 2-1 Yamadaoka, Suita, Osaka, Japan

*E-mail: ohtsuka@mech.eng.osaka-u.ac.jp
キーワード：実時間最適化(real-time optimization)，receding horizon 制御 (receding horizon control)，モデル予測制御 (model predictive control)．
JL 0005/02/4105-0366 © 2002 SICE

1. はじめに

Receding Horizon (RH) 制御とは，各時刻において有限時間未来までの応答を最適化する制御手法である．最適制御問題の評価区間 (horizon) が時間とともに移動していることになるので，receding horizon 制御または moving horizon 制御と呼ばれる．同様の手法はプロセス制御分野を中心に以前から用いられており，モデル予測制御とも呼ばれる[1]．従来は主として線形系を制御対象とした研究が行われてきたが，近年非線形系への適用が盛んに研究されている[2),3)]．非線形 RH 制御の課題には，大きく分けると閉ループ系の解析と実時間アルゴリズムがある．本稿では，速いダイナミクスをもつ機械系などへの応用を目指して筆者が開発した実時間アルゴリズムを紹介する．

2. Receding Horizon 制御

最初に，RH 制御を明確に定式化しておく．以後，扱う関数は必要なだけ微分可能とする．制御対象は以下の状態方程式で表わされるとする．

$$\dot{x}(t)=f(x(t),u(t))$$

ここで，$x(t)\in\mathbf{R}^n$ は状態ベクトル，$u(t)\in\mathbf{R}^{m_u}$ は制御入力ベクトルである．初期状態 $x(0)=x_0$ から出発して制御入力関数 $u(\cdot)$ が与えられたとき，時刻 t における状態を $x^u(t;x_0)$ と表わすことにする．制御入力関数は区分的連続とする．また，状態や入力に関するさまざまな拘束条件が

$$C(x(t),u(t))=0$$

なる m_c 次元の等式拘束として課せられているとする．たとえば，不等式拘束の場合，ダミー変数（スラック変数）を入力 u に含めることで等式拘束に変換できる．

まず，評価区間が固定された次の評価関数を考える．

$$J(x_0,T,u(\cdot)):=\varphi(x^u(T;x_0))+\int_0^T L(x^u(t;x_0),u(t))dt$$

これを最小にする最適制御 $u^*(\cdot;x_0,T)$ が存在したとしても，それは時刻 T までの制御入力しか与えない．しかし，実際のフィードバック制御では，ある固定された時刻までのみの制御性能を評価する代わりに，つねに一定時間未来までの制御性能を評価しつづける，という考え方もありうる．それが RH 制御である．状態 x から T だけ未来までを評価したときの最適制御入力とは，最適制御の初期値 $u^*(0;x,T)$ に他ならない．したがって，RH 制御の最適状態フィードバック制御則 $u^{RH}(x)$ は次式で与えられる．

$$u^{RH}(x):=u^*(0;x,T)$$

これは，T が固定ならば時不変な状態フィードバック制御則になっている．

ただし，一定時間未来までの応答を最適化したとしても，閉ループ系の安定性が保証されるとは限らない．また，閉ループ系の実際の軌道は，評価区間が固定された最適制御問題の最適軌道とは必ずしも一致しない．そこで，閉ループ系の解析が近年盛んに研究されている[4),5)]．これらの研究が進めば，安定性や性能を保証するためにどのような評価関数や拘束条件を選べばよいかが明らかになると期待される．

3. アルゴリズム

3.1 概観

つぎに非線形 RH 制御のアルゴリズムを概観する．アルゴリズムを大別すると，オンラインで各時刻の状態 $x(t)$ に対する制御入力 $u^{RH}(x(t))$ を計算する方法と，オフラインで状態フィードバック制御則 $u^{RH}(x)$ を状態 x の関数として計算して保存しておく方法とがある．

オフラインでの計算方法は主としてハミルトン・ヤコビ・ベルマン方程式 (HJBE) の解法に帰着される．紙面の都合で紹介できないが，HJBE の解法に関して興味深い研究が近年数多くなされている．ただし，状態 x の関数を保存する場合，状態空間の次元が大きくなるにつれ，記憶量の急激な増大（次元の呪い）が問題となる．

オンラインの計算方法では，各時刻の状態 $x(t)$ に対応する最適制御 $u^*(\cdot;x(t),T)$ を時間関数として求め，$u^{RH}(x(t)):=u^*(0;x(t),T)$ とする．したがって，勾配法など通常の最適制御問題のさまざまな数値解法[6)~8)]が適用できる．また，非線形 RH 制御問題におけるヘシアンなどの構造を利用して効率化を図った最適化アルゴリズムが提案されつつある[3),9)]．しかし，最適化手法として通常の反復計算を用いる限り，2 次計画問題や連立 1 次方程式を解いての探索方向決定と直線探索とを繰り返さなくてはならない．特に最適制御問題の場合，評価関数の値を計算するには状態方程式を評価区間にわたって積分（つまりシミュ

レーション）しなければならないから，直線探索1つ取っても計算量は無視できない．

ところが，最適制御 $u^*(\cdot\,;x(t),T)$ 自体が $x(t)$ を通じて時間依存であることを利用して，その時間変化を追跡すれば，反復計算は必ずしも必要ではない．つまり，時刻 t をパラメータとする一種の連続変形法（ホモトピー法）[10]である．連続変形法をRH制御問題に適用すると，最適解（厳密には停留解）の時間変化を表わす微分方程式が近似なしに得られるので，それを実時間で数値積分するアルゴリズムが考えられる[11)~13)]．しかし，近似なしに得られた微分方程式は複雑なので，近似を導入してさらに計算量を減らす工夫がなされている[14)]．以下では近似を導入した手法を紹介する．

3.2 離散化された問題

最適制御問題の評価区間を N ステップに分割して離散近似された問題を各時刻で解くことにする．離散近似された問題に対する最適性の必要条件は，以下のような2点境界値問題になる．

$$x_{i+1}^*(t)=x_i^*(t)+f(x_i^*(t),u_i^*(t))\Delta\tau \quad (1)$$

$$x_0^*(t)=x(t) \quad (2)$$

$$H_u(x_i^*(t),\lambda_{i+1}^*(t),u_i^*(t),\mu_i^*(t))=0 \quad (3)$$

$$C(x_i^*(t),u_i^*(t))=0 \quad (4)$$

$$\lambda_i^*(t)=\lambda_{i+1}^*(t)+H_x^T(x_i^*(t),\lambda_{i+1}^*(t),u_i^*(t),\mu_i^*(t))\Delta\tau \quad (5)$$

$$\lambda_N^*(t)=\varphi_x^T(x_N^*(t)) \quad (6)$$

ここで，$\Delta\tau:=T/N$ であり，$x_i^*(t)\in\mathbf{R}^n$ は $x(t)$ を初期状態とする離散近似最適制御問題の i ステップめの状態を表わし（図1），$\lambda_i^*(t)\in\mathbf{R}^n$, $u_i^*(t)\in\mathbf{R}^{m_u}$, $\mu_i^*(t)\in\mathbf{R}^{m_c}$ はそれぞれ共状態，制御入力，拘束に関するラグランジュ乗数である．また，H は以下で定義されたハミルトニアンである．

$$H(x,\lambda,u,\mu):=L(x,u)+\lambda^Tf(x,u)+\mu^TC(x,u)$$

一般に評価区間の長さ T は時刻 t に依存し，したがって $\Delta\tau$ も同様である．より一般に問題自体が時変でも構わない．各時刻 t において，測定された状態 $x_0^*(t)=x(t)$ に対して(1)～(6)式の2点境界値問題を解き，実際の制御入力は $u(t)=u_0^*(t)$ で与える．問題全体が時刻 t に依存し，また，t に関しては離散化されていないことに注意されたい．

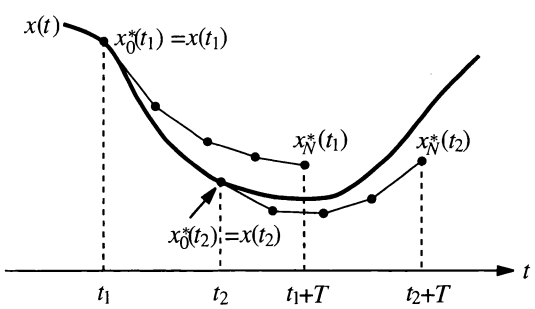


図1 各時刻で計算する離散近似問題の最適軌道 $\{x_i^*(t)\}_{i=0}^{i=N}$ と実際の連続時間閉ループ軌道 $x(t)$ との関係

制御入力とラグランジュ乗数の系列をまとめたベクトル $U(t)$ および射影 P_0: $\mathbf{R}^{mN}\to\mathbf{R}^{m_u}$ を以下のように定義する．

$$U(t):=[u_0^{*T}(t)\ \mu_0^{*T}(t)\ u_1^{*T}(t)\ \mu_1^{*T}(t)\cdots u_{N-1}^{*T}(t)\ \mu_{N-1}^{*T}(t)]^T\in\mathbf{R}^{mN},$$

$$P_0(U(t)):=u_0^*(t)$$

ただし，$m:=m_u+m_c$ である．$U(t)$ と $x(t)$ が与えられると，$\{x_i^*(t)\}_{i=0}^N$ と $\{\lambda_i^*(t)\}_{i=0}^N$ は(1)，(2)，(5)，(6)式から決まるので，(3)，(4)式はつぎのような mN 次元の方程式と見なせる．

$$F(U(t),x(t),t):=\begin{bmatrix} H_u^T(x_0^*(t),\lambda_1^*(t),u_0^*(t),\mu_0^*(t)) \\ C(x_0^*(t),u_0^*(t)) \\ \vdots \\ H_u^T(x_{N-1}^*(t),\lambda_N^*(t),u_{N-1}^*(t),\mu_{N-1}^*(t)) \\ C(x_{N-1}^*(t),u_{N-1}^*(t)) \end{bmatrix}=0$$

各時刻で測定した $x(t)$ に対して上の方程式を解き，$U(t)$ を求めれば，制御入力が $u(t)=P_0(U(t))$ として決まる．したがって，状態フィードバック制御則が陰に定められていることになる．

3.3 解の追跡

各時刻において方程式 $F(U,x,t)=0$ をニュートン法のような反復解法で解くのは非効率的なので，その代わりに $F=0$ が安定となるように $U(t)$ を更新していくことを考える．つまり，次式が成り立つように $\dot{U}$ を決定する．

$$\dot{F}(U,x,t)=-\zeta F(U,x,t) \quad (7)$$

ここで ζ は正の実数である．ヤコビアン F_U が正則であれば，上の条件から得られる $U(t)$ の微分方程式

$$\dot{U}=F_U^{-1}(-\zeta F-F_x\dot{x}-F_t) \quad (8)$$

を実時間で積分することにより $U(t)$ を更新していくことができる．もし $F(U(0),x(0),0)=0$ が成り立つように $U(0)$ を選べれば，時刻とともに変化する解 $U(t)$ が追跡されることになり，一種の連続変形法となる．たとえば，評価区間の長さ T の初期値を0として2点境界値問題の自明な解から出発し，その後 T を滑らかに増加させていくことや，制御開始前に時間をかけて勾配法などで初期解 $U(0)$ を求めることが考えられる．

微分方程式(8)から $\dot{U}(t)$ を決定するには，ヤコビアン F_U,F_x,F_t を評価しなければならず，また，F_U^{-1} が含まれているため連立1次方程式を解く必要もある．これらの行列のサイズは評価区間の分割数 N に応じて巨大になりうるので，効率の良い計算方法が必要である．そこで，連立1次方程式の解法としてGMRES（Generalized Minimum RE Sidual）法[15,16)]を採用し，さらにヤコビアンとベクトルの積を前進差分で近似する．

まず，$\dot{U}$ の条件を書きなおした

$$F_U\dot{U}=-\zeta F-F_x\dot{x}-F_t \quad (9)$$

は，$\dot{U}$ に関する連立1次方程式と見なせるので，GMRES法を適用できる．GMRES法は，共役勾配法を含むクリロフ部分空間法の一種であり，非対称正方行列を係数にもつ連立1次方程式の反復解法として1986年に提案された．たとえば $Ax=b$ なる連立1次方程式に対して，初期推定解を x_0，初期残差を $r_0:=b-Ax_0$ とするとき，GMRESの k 回めの反復では，

$$\min_{x\in x_0+K_k}\|b-Ax\| \tag{10}$$

なる最小2乗問題の解 x が計算される．ここで，$K_k:=\mathrm{span}\{r_0, Ar_0, \cdots, A^{k-1}r_0\}$ をクリロフ部分空間と呼ぶ．GMRESのおもな特徴をまとめると以下の通りである．

(i) (10)式の最小2乗問題が特殊な形に帰着できるため効率的に解ける．

(ii) 各反復において行列 A 自体は必要なく，行列 A と K_k の正規直交基底に属するベクトル v_k との積 Av_k を1回だけ計算すればよい．

(iii) 理論的には未知変数ベクトルの次元に等しい回数の反復で収束が保証されるが，実際にはより少ない反復で十分な精度の解が得られることが多く，大規模な問題に対して有効とされている．

RH制御への適用では特に2番目の特徴が有効で，連立1次方程式(9)の係数行列であるヤコビアン F_U を直接求める必要がなくなり，計算量が大幅に低減できる．すなわち，あるベクトル $W\in\mathbf{R}^{mN}$, $w\in\mathbf{R}^n$ および $\omega\in\mathbf{R}$ に対し，それらとヤコビアンとの積を，つぎのように方向微分の前進差分で近似する[15]．

$$F_U(U,x,t)W+F_x(U,x,t)w+F_t(U,x,t)\omega \simeq \frac{F(U+hW,x+hw,t+h\omega)-F(U,x,t)}{h}$$

ここで，h は正の実数である．関数 F を2回評価するだけでヤコビアンとベクトルの積を近似できることに注意されたい．関数 F を評価するには状態量と共状態量の方程式(1)，(5)を評価区間にわたって解かなければならないので，F の評価回数を減らすことは計算量低減に有効である．

また，RH制御を実装する際には(9)式を各サンプリング時刻ごとに解くので，直前のサンプリング時刻における $\dot{U}$ をGMRESの初期推定解として用いることができる．したがって，3番目の特徴ともあいまって，$\dot{U}$ の次元が数百でもわずか数回の反復で十分な精度が得られる場合が多い．

連続変形 (Continuation) 法とGMRES法を組み合わせたアルゴリズム (C/GMRES) をまとめると，以下のようになる．

1. 評価区間の長さ $T(t)$ を，$T(0)=0$ かつ $T(t)\to$ const. $(t\to\infty)$ を満たす滑らかな関数に選ぶ．初期状態 $x(0)$ を測定し，$x_i^*(0)=x(0)$, $\lambda_i^*(0)=\varphi_x^T(x(0))$ $(i=0,\cdots,N)$ とする．また，正の実数 δ を選び，

$$\left\|\begin{bmatrix} H_u^T(x(0),\varphi_x^T(x(0)),u(0),\mu(0)) \\ C(x(0),u(0)) \end{bmatrix}\right\|\le\frac{\delta}{\sqrt{N}}$$

を満たす $u(0)$, $\mu(0)$ を解析的もしくは数値的に求め，$u_i^*(0):=u(0)$, $\mu_i^*(0):=\mu(0)$ $(i=0,\cdots,N-1)$ とする．これによって，$\|F(U(0),x(0),0)\|\le\delta$ を満たす初期条件 $U(0)$ が得られる．

2. 各時刻 t において $x(t)$ を測定し，(9)式にGMRESを適用して得られる $\dot{U}(t)$ を実時間で数値積分して $U(t)$ を求める．制御入力は $u(t)=P_0(U(t))$ で与える．

C/GMRESの特徴をまとめると以下のようになる．

(i) 離散化された問題(1)～(6)を解いているので，計算時間等の制約から離散近似の精度を犠牲にして評価区間の分割数 N を少なくしても，計算自体は数値的困難に陥ることなく実行できることが多い．

(ii) 拘束条件を容易に扱える．特に，入力を含まない状態量拘束に対しても特別な扱い[17]が必要ない．これは，$C_u=0$ でも F_U が正則になるためである．

(iii) $H_u=0$ を陽に解くことや，高次の偏導関数 H_{uu} や H_{xx} が必要ないので，状態方程式や評価関数が複雑になってもプログラミングが容易で，計算量の増大も抑えられる．

(iv) 問題のサイズ（未知量 U の次元）は状態の次元によらず，入力と拘束の次元 m と分割数 N との積 mN で与えられる．

(v) 反復計算はGMRES内部で行われるのみであり，連立1次方程式を各時刻で複数回解く必要はない．直線探索も必要ない．

(vi) 上記の特徴により，GMRES内部での反復回数を固定しておけば，$U(t)$ の更新に要する時間はサンプリング時刻によらず一定である．

C/GMRESの代わりに，たとえばニュートン法を用いる場合，U の修正量はやはり連立1次方程式で決まるが，各時刻ごとに U の修正を反復して収束させるため，連立1次方程式を複数回解かなければならない．また，収束するまでの反復回数が各サンプリング時刻でばらついてしまうと，一定サンプリング周期での実装には適さない．

4. 数値例と適用のポイント

4.1 プログラムの自動生成

2リンク・アームの振り上げ制御を題材に，C/GMRESを適用する際の留意点を述べる．制御対象はロボット・アームとしては単純であり4状態2入力だが，複雑な非線形状態方程式で記述される．シミュレーション・プログラムは，AutoGenU[18]で自動生成した．AutoGenUは数式処理言語Mathematicaで記述されたプログラムであり，状態方程式や評価関数を定義する入力ファイルを読み込んで偏微分計

算などを行い，C/GMRESによる非線形RH制御のシミュレーション・プログラムをC言語のソース・ファイルとして自動的に生成する．図2に入力ファイルの例を示す．fxuが状態方程式を，Lとphiが評価関数を，それぞれ定義している．ユーザは適当なテキストエディタを用いて入力ファイルを作成するだけでよい．状態方程式や評価関数はMathematicaの書式で書かれているので，複雑な機械系の場合は状態方程式もMathemticaで導出すれば労力が省ける．

4.2 計算時間

シミュレーションの時間刻みを5 msとして，1.5 sの制御をシミュレーションした．評価区間の長さは，$T(0)=0$ かつ $T(t)\to$const.$(t\to\infty)$ となるよう $T(t):=T_f(1-e^{-\alpha t})$, $T_f=0.2$[s], $\alpha=1.5$ とした．用いた計算機は，CPU: Pentium II，300 MHzのPCである．**表1**は，いくつかのアルゴリズムについて，制御入力更新の平均計算時間と分割数当たりの誤差 $\|F\|/N$ の平均値を示している（ただし，Backward Sweepの誤差は定義が異なる）．C/GMRESの場合は制御入力更新の計算時間は一定なので，表1の平均計算時間がそのまま実現可能なサンプリング周期となる．他のアルゴリズムは，Backward Sweepが未知共状態量を更新していくアルゴリズム[13]，また，L-BFGS[19]とTruncated Newton[20]はそれぞれ汎用の最適化手法を適用した例である．表から，C/GMRESは他より高速でかつ計算時間が分割数 N に関して線形にしか増加しないことが分かる．なお，C/GMRES内部におけるGMRESの反復は，N によらずわずか2回で十分だった．

```
(*-------- Define dimensions of x, u, C(x,u), and p(t). --------*)
dimx=4;
dimu=2;
dimc=0;
dimp=0;
(*-------- Do not touch the following difinition of vectors. --------*)
xv=Array[x,dimx];
lmdv=Array[lmd,dimx];
uv=Array[u,dimu];
muv=Array[u,dimc,dimu+1];
pv=Array[p,dimp];
(*-------- Define f(x,u), C(x,u), p(t), L(x,u) and phi(x). --------*)
fxu = {x[3], x[4], (-ga*J2*(LC1*m1 + L1*m2)*Sin[x[1]] +
   L1*LC2*m2*Cos[x[2]]*(ga*LC2*m2*Sin[x[1] + x[2]] - u[2] +
   L1*LC2*m2*Sin[x[2]]*x[3]^2) +
   J2*(u[1] - u[2] + L1*LC2*m2*Sin[x[2]]*x[3]^2 +
   2*L1*LC2*m2*Sin[x[2]]*x[3]*x[4] + L1*LC2*m2*Sin[x[2]]*x[4]^2))/
   (J2*(J1 + L1^2*m2) - L1^2*LC2^2*m2^2*Cos[x[2]]^2),
   ((J1 + J2 + L1^2*m2 + 2*L1*LC2*m2*Cos[x[2]])*
   (-ga*LC2*m2*Sin[x[1] + x[2]] + u[2] - L1*LC2*m2*Sin[x[2]]*x[3]^2) \
   + (-J2 - L1*LC2*m2*Cos[x[2]])*
   (-ga*(LC1*m1 + L1*m2)*Sin[x[1]] - ga*LC2*m2*Sin[x[1] + x[2]] +
   u[1] + L1*LC2*m2*Sin[x[2]]*x[4]*(2*x[3] + x[4])))/
   (J2*(J1 + L1^2*m2) - L1^2*LC2^2*m2^2*Cos[x[2]]^2)};
Cxu={};
pt={};
qv=Array[q,dimx];
rv=Array[r,dimu];
sfv=Array[sf,dimx];
xfv=Array[xf,dimx];
Q=DiagonalMatrix[qv];
R=DiagonalMatrix[rv];
Sf=DiagonalMatrix[sfv];
L = (xv - xfv).Q.(xv - xfv)/2 + uv.R.uv/2;
phi = (xv - xfv).Sf.(xv - xfv)/2;
(*-------- Define user's variables and arrays. --------*)
(*--- Numbers must be in Mathematica format. ---*)
MyVarNames = {"ga", "m1", "L1", "LC1", "J1", "m2", "L2", "LC2", "J2"};
MyVarValues = {9.80665, 0.2, 0.3, 0.15, 0.2*0.15^2 + 1.5*10^(-3), 0.7,
   0.3, (0.2*0.15+0.5*0.3)/0.7, 0.2*0.15^2+1.5*10^(-3)+0.5*0.3^2};
MyArrNames = {"q", "r", "sf", "xf"};
MyArrDims = {dimx, dimu, dimx, dimx};
MyArrValues = {{40, 20, 0.01, 0.01}, {1,1}, {4, 2, 0.001, 0.001},
   {3.14156, 0, 0, 0}};
```

図2 入力ファイルの例

表1 制御入力更新の平均計算時間[ms]（上段）と分割数当たりの誤差 $\|F\|/N$ の平均（下段）

N	C/GMRES	Backward Sweep	L-BFGS	Truncated Newton
5	0.9 3.12 E-03	Failure	1.8 2.74 E-03	1.4 1.16 E-03
10	1.6 3.04 E-03	Failure	4.6 3.19 E-03	3.1 2.35 E-03
20	3.3 2.82 E-03	Failure	17.7 2.72 E-03	26.2 2.71 E-03
50	7.9 2.32 E-03	Failure	48.9 8.31 E-03	63.9 8.92 E-03
100	15.4 1.86 E-03	34.8 (0.1113)	96.3 1.84 E-02	211.2 2.20 E-02

4.3 拘束の扱い

つぎに，拘束条件の具体的な扱い方について簡単に述べる．たとえば，大きさが $|u_1|\le u_{1\,max}$ と拘束された肩関節トルク u_1 のみによる振り上げ制御を考える．この場合，ダミー入力 u_3 を導入し，不等式拘束を等式拘束

$$C(u_1, u_3):=u_1^2+u_3^2-u_{1\,max}^2=0$$

に変換すればC/GMRESが適用できる．状態量不等式拘束の場合も同様である．ただし，注意しなければならないのは，ダミー入力 u_3 は2次項でしか現れないので，ダミー入力の符号が正負どちらでも最適解になりうることである．したがって，ダミー入力の符号が最適性の条件から決まらず計算に失敗してしまう場合がある．それを避けるには，ダミー入力の符号として正負どちらが望ましいかを評価関数にわずかでも含めておけばよいので，たとえば，r_3 を小さい正数として，$-r_3u_3$ などという項を評価関数の被積分項 L に付け加える．そのようにして計算した例が**図3**である．$|u_1|\le 4$[N·m] という拘束を守りつつアームの振り上げが行われている．

なお，制御入力不等式拘束付き問題の多くでは，評価関数に制御入力が含まれなければバンバン制御が最適になるが，C/GMRESでは解の追跡ができない．そのような場合は，制御入力の小さなペナルティを評価関数に付加して近似的に計算する．その他，IF～THENルールを含む問題も近似的に滑らかな問題として扱う工夫が考えられる．

4.4 パラメータの選び方など

アルゴリズムに含まれるさまざまなパラメータをどう選ぶかも実際の適用では重要なので整理しておく．

評価区間の分割数 N は多いほうが元の連続時間最適制御問題をよく近似できるが，入力更新の計算時間は N に比例する．サンプリング周期 Δt に比べて評価区間の刻み $T/$

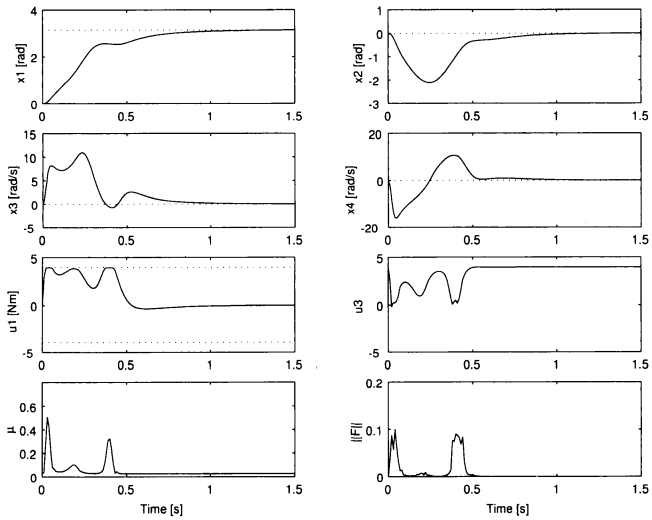


図3　拘束された肩関節トルクのみによる2リンク・アームの振り上げ

Nが10倍程度大きくても十分な制御性能が得られることも多い．

安定化パラメータζに関しては，GMRESの残差や数値積分の効果も考慮した詳細な誤差解析[14]の結果，$\dot{U}(t)$をオイラー法で数値積分したとき，いくつかの仮定の下で，$\zeta=1/\Delta t$と選べばつねに$F=0$が安定であることがわかっている．経験的にも，試行錯誤せずに$\zeta=1/\Delta t$と選べばよいようである．

GMRESの反復回数k_{max}は，数回程度で十分なことが多い．$\dot{U}$の変化が激しい問題では，大きめにする必要がある．

ヤコビアン前進差分の刻みhは理論上小さいほどよいが，丸め誤差との兼ね合いであまり小さくはできない．今のところ明確な決定方法はない．

サンプリング周期Δtは当然C/GMRESによる入力更新の計算時間より長くなければならない．C/GMRESの計算時間はNとk_{max}におおよそ比例するので，それらによって決まることになる．

一方，評価関数をどのように選ぶかも重要である．基本的には，φ, LともにLQ制御のような2次形式で与えて，その重み行列を調節すればよい．RH制御で評価区間が短い場合，φは応答に大きく影響するから重要である．また，状態量の重みが極端に大きいと数値的困難に陥ることがあるので，状態量の重みは小さい値から徐々に増やしてシミュレーションを繰り返し，応答を改善していく．逆に，入力の重みはある程度大きくないと計算できない．

評価区間の長さを0からはじめる場合は，先ほどの例題のように$T=T_f(1-e^{-\alpha t})$とすればよい．最終的な評価区間の長さT_fが大きすぎると計算に失敗する．機械系では1秒前後が妥当なようである．αは，あまり大きいとTの増加が急すぎて解の追跡に失敗するので，0.5～1.5程度に選ぶ．

一般に，評価関数の決定には，制御対象の性質と評価関数の意味に対する洞察が重要である．現時点では試行錯誤に頼ることが多く，体系的な決定方法は手付かずの問題である．実際の応用によるノウハウの蓄積と共に，閉ループ系解析のさらなる理論的発展を待つ部分も大きい．

最後に，ハードウェア実装時の重要なポイントは，センサのノイズをある程度除去しておくことである．C/GMRESは，つねに誤差Fを減少させるよう計算していくので，ある程度の外乱やノイズが加わっても計算に失敗することはないが，ノイズがあまりに大きいと解の追跡に失敗することがある．特に，位置を差分して速度を求めている場合は注意が必要である．

5.　おわりに

本稿では，非線形RH制御の実時間アルゴリズムについて実際的な観点から解説した．ここで述べた手法は，非線形RH制御に限らず，時間や状態に依存する代数方程式や最適化問題をオンラインで解く用途にも適用可能である．本稿が実際の問題を解くのに役立てば幸いである．今後，より複雑でより大規模な問題をより高速に解くアルゴリズムが見つかれば，それだけ実時間で解ける問題のクラスが広がるので，たとえ計算機の能力が向上しても，アルゴリズムの研究には意義があるだろう．

(2002年1月31日受付)

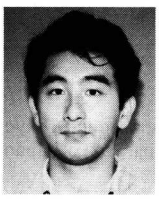
参　考　文　献

1) ミニ特集：予見・予測制御，計測と制御，**39**-5 (2000)
2) M. A. Henson and D. E. Seborg (eds.) : Nonlinear Process Control, Prentice Hall (1997)
3) F. Allgöwer and A. Zheng (eds.) : Nonlinear Model Predictive Control, Birkhäuser (2000)
4) D. Q. Mayne, J. B. Rawlings, C. V. Rao and P. O. M. Scokaert : Constrained Model Predictive Control : Stability and Optimality, Automatica, **36**-6, 789/814 (2000)
5) F. A. C. C. Fontes : A General Framework to Design Stabilizing Nonlinear Model Predictive Controllers, Systems & Control Letters, **42**, 127/143 (2001)
6) 嘉納：システムの最適理論と最適化，コロナ社 (1987)
7) J. T. Betts : Practical Methods for Optimal Control Using Nonlinear Programming, SIAM (2001)
8) 井前：非線形最適制御問題の数値解法―リッカチ微分方程式を中心として，システム/制御/情報，**46** (2002) 掲載予定
9) R. L. Tousain and O. H. Bosgra : Efficient Dynamic Optimization for Nonlinear Model Predictive Control, Proceedings of the 39 th IEEE Conference on Decision and Control, Sydney, Australia, 760/765 (2000)
10) S. L. Richter and R. A. DeCarlo : Continuation Methods : Theory and Applications, IEEE Transactions on Automatic Control, **AC-28**-6, 660/665 (1983)
11) 大塚：非線形最適フィードバック制御のための実時間最適化手法，計測と制御，**36**-11, 776/783 (1997)
12) T. Ohtsuka and H. A. Fujii : Real-Time Optimization Algorithm for Nonlinear Receding-Horizon Control, Automatica, **33**-6, 1147/1154 (1997)
13) T. Ohtsuka : Time-Variant Receding-Horizon Control of

Nonlinear Systems, Journal of Guidance, Control, and Dynamics, 21-1, 174/176 (1998)
14) T. Ohtsuka : Continuation/GMRES Method for Fast Algorithm of Nonlinear Receding Horizon Control, Proceedings of the 39 th IEEE Conference on Decision and Control, Sydney, Australia, 766/771 (2000)
15) C. T. Kelley : Iterative Methods for Linear and Nonlinear Equations, Vol. 16 of Frontiers in Applied Mathematics, SIAM (1995)
16) 藤野, 張：反復法の数理, 朝倉書店 (1996)
17) 加藤：工学的最適制御, 東京大学出版会 (1988)
18) T. Ohtsuka : http://www-newton.mech.eng.osaka-u.ac.jp/~ohtsuka/code/index.htm (2000)
19) J. Nocedal : http://www.netlib.org/opt/lbfgs_um.shar (1990)
20) S. G. Nash : http://www.netlib.org/opt/tn (1984)

[著 者 紹 介]

大塚敏之君 (正会員)



1967年9月28日生. 95年3月東京都立科学技術大学大学院博士課程(工学システム専攻)修了. 同年4月筑波大学構造工学系講師. 99年4月大阪大学大学院工学研究科講師. 96年9月～97年2月カリフォルニア工科大学客員研究員. 主として航空宇宙工学および機械工学に応用をもつ制御理論の研究に従事. 博士(工学). 日本航空宇宙学会, 日本機械学会, システム制御情報学会, AIAA, IEEEの会員.