포물선 근사법에 의한 상태방정식의 새로운 수치해석적 접근법에 관한 연구

이 종기, 권 용준, 최 병곤, 문 영현
연세대학교

A New Numerical Method for Solving Differential Equation by Quadratic Approximation

Jong-Gi Lee, Yong-Jun Kwon, Byoung-Kon Choi, Young-Hyun Moon
Dept. of Electical Engineering, Yonsei Univ.

Abstract - 전력계통의 과도 안정도 해석의 접근방법에는 SI(Simultaneous Implicit)법과 PE(Partitioned Explicit)법 두 가지방법을 사용해오고 있다. SI법에는 Trapezoidal법 등이 있고, PE법에는 Runge-Kutta법, Euler법등이 사용되고 있다. SI법인 Trapezoidal법은 PE법의 Runge-Kutta법 또는 Euler법에 비해 시간간격을 크게 해서 계산속도를 줄일 수 있다는 장점이 있지만, 정확도면에서는 신뢰할 수 없는 단점이 있다. 이 논문에서는 포물선 근사법을 이용하여 Trapezoidal법의 정확도를 개선할 수 있는 방법을 제시하고 명확한 수학적 증명을 통해 타당성을 보여준다. 연속함수와 불연속함수에 대해서 Runge-Kutta법과 Trapezoidal법과 제안한 방법을 적용시켜서 제안한 방법의 정확함을 보여준다.

1. 서 론

전력계통은 비선형 미분방정식이며 상태변수의 업데이트에 대한 관점에서 본다면, 현재 다양한 알고리즘이 쓰이고 있는데 크게 PE법과 SI법이다.

대부분의 비선형 시스템의 해를 구하는 정확한 방법이 존재하지 않기 때문에, 많은 노력들이 수치해석적인 방법에 집중되어왔다. 미분방정식의 해를 구하는 수치해석적인 방법에는 크게 SI법과 PE법이 있다. 비선형 시스템 시뮬레이션의 Implicit 방법의 안정도가 조사되었다[1]. Neural Network와 Riccati differential equation에도 몇몇 수치해석적인 방법이 적용되어 왔다[2,3,4].

이 논문에서는 미분방정식의 해를 구하는 새로운 방법을 제시하고, 수학적으로 증명을 하였다. Trapezoidal 방법에서는, 함수를 부분적으로 직선으로 근사화하여 사다리꼴을 적분하므로써 해를 구하기 때문에 오차가 발생한다. 이 논문에서는 Trapezoidal방법의 오차를 줄이기 위해 적분구간을 직선이 아닌 2차곡선으로 근사화하여 적분을 한다.

이 논문에서 제안된 방법과 Trapezoidal법과 Runge-Kutta법을 적용하여 결과를 비교하였다. 불연속이 포함된 경우에 있어서 제안된 방법이 Trapezoidal법이나 Runge-Kutta법보다 더 정확한 결과를 보여준다.

2. 수학적 분석

2.1 제안된 방법의 수학적 분석

Trapezoidal법에서는 사다리꼴의 면적을 더하면서 적분을 한다. 이 때에 적분구간을 직선으로 근사화하게된다. 제안된 방법을 적분구간을 2차함수로 근사화함으로써 그 오차를 줄일 수 있다. 식(1)의 미분방정식이 주어진다.

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), t) \tag{1}$$

식(1)의 미분방정식을 다음과 같이 풀 수 있다.

$$\mathbf{x}(t_i + \Delta T) - \mathbf{x}(t_i) = \int_{t_i}^{t_i + \Delta T} \mathbf{f}(\mathbf{x}(t), t)dt = \int_{t_i}^{t_i + \Delta T} \mathbf{g}(t)dt \tag{2}$$

여기서, $\mathbf{g}(t) = \mathbf{f}(\mathbf{x}(t), t)$ (2')

식(2)의 우변을 Taylor급수로 전개하면 다음과 같다.

$$\mathbf{g}(t) = \mathbf{g}(t_i) + \bar{\mathbf{m}}\Delta t + \bar{\mathbf{a}}\Delta t^2 \tag{3}$$

위 식(3)에서 m과 a를 구하려면, 다음의 식(4), (5)를 이용할 수 있다.

$$\text{i)}\ \bar{\mathbf{m}} = \frac{d\mathbf{g}(t)}{t} = \frac{d}{dt}\mathbf{f}(\mathbf{x}(t), t)|_{t\ t_i} \frac{\partial \mathbf{f}(\mathbf{x}(t), t)}{\partial \mathbf{x}(t)}|_{t\ t_i} \cdot \mathbf{f}(\mathbf{x}(t_i), t_i) \frac{\partial \mathbf{f}(\mathbf{x}(t), t)}{\partial \mathbf{x}(t)}|_{t=t_i} \tag{4}$$

$$\text{ii)}\ \mathbf{g}(t_i + \Delta T) \approx \mathbf{g}(t_i) + \bar{\mathbf{m}}\Delta T + \bar{\mathbf{a}}\Delta T^2$$

$$\therefore \bar{\mathbf{a}} = \frac{1}{\Delta T^2}\left(\mathbf{g}(t_{i+1}) - \mathbf{g}(t_i) - \bar{\mathbf{m}}\Delta T\right) = \frac{1}{\Delta T^2}\left[\mathbf{f}(x_{i+1}, t_{i+1}) - \mathbf{f}(x_i, t_i) - \bar{\mathbf{m}}\Delta T\right] \tag{5}$$

식(4), (5)를 (3)에 대입하면 식(6)을 구할 수 있다.

$$\mathbf{g} = \mathbf{g}(t_i) + \bar{\mathbf{m}}\Delta t + \frac{1}{\Delta T^2}\left(\mathbf{g}(t_{i+1}) - \mathbf{g}(t_i) - \bar{\mathbf{m}}\Delta T\right)\Delta t^2 \tag{6}$$

식(6)을 식(2)에 대입하면 식 (7)과 같이 된다.

$$\int_{t_i}^{t_i + \Delta T} \mathbf{g}(t) = \int_0^{\Delta T}\left[\mathbf{g}(t_i) + \bar{\mathbf{m}}\Delta + \frac{1}{\Delta T^2}\left\{\mathbf{g}(t_{i+1}) - \mathbf{g}(t_i)\bar{\mathbf{m}}\Delta T\right\}\Delta t^2\right]d\Delta t$$

$$= \mathbf{g}(t_i)\Delta T + \frac{1}{2}\bar{\mathbf{m}}\Delta T^2 + \frac{1}{3}\Delta T\left[\mathbf{g}(t_{i+1}) - \mathbf{g}(t_i) - \bar{\mathbf{m}}\Delta T\right]$$

$$= \frac{1}{3}\left[\mathbf{g}(t_{i+1}) + 2\mathbf{g}(t_i)\right]\Delta T + \frac{1}{6}\bar{\mathbf{m}}\Delta T^2 \tag{7}$$

식 (7)을 식(2)에 대입하고 (2')의 관계를 이용해서 다시 정리하면 다음과 같은 식 (8)을 얻을 수 있다.

$$x(t_{i+1}) - \frac{1}{3}\Delta T f(x(t_{i+1}), t_{i+1}) = x(t_i) + \frac{2}{3}\Delta T f(x(t_i), t_i) + \frac{1}{6}\bar{m}(t_i)\Delta T^2 \quad (8)$$

$$x^{(k+1)} - \frac{1}{3}\Delta T f(x^{(k+1)}, t^{(k+1)}) = x^{(k)} + \frac{2}{3}\Delta T f(x^{(k)}, t^{(k)}) + \frac{1}{6}\bar{m}(t^{(k)})\Delta T^2 \quad (9)$$

식(9)의 우변을 Constant로 놓고 식을 정리하면 식(10)을 얻을 수 있다.

$$F = x - \frac{1}{3}\Delta T f - C \quad (10)$$

2.2 미분방정식

다음의 미분방정식(12)를 고려해보자. 실제 해는 식(13)에 주어져있다.

$$\dot{x_1} = x_2$$
$$\dot{x_2} = x^2{}_2 - e^{-x_1}\sin t \qquad x_1(0) = 0, x_2(0) = 2 \quad (12)$$

$$\begin{cases} x_1 = \log(1 + t + \sin t) \\ x_2 = \dfrac{1 + \cos t}{1 + t + \sin t} \end{cases} \quad (13)$$

식(4)로부터 m을 구하면 식(14)와 같다.

$$\bar{m}\frac{dg(t)}{t} = \frac{d}{dt}f(x(t), t)|_{t=t_i}$$
$$\frac{\partial f(x(t), t)}{\partial x(t)}|_{t=t_i} \cdot f(x(t_i), t_i)\frac{\partial f(x(t), t)}{\partial x(t)}|_{t=t_i} \quad (14)$$
$$= \begin{bmatrix} -x^2{}_2 - e^{-x_1}\sin t \\ x_2 e^{-x_1}\sin t - 2x_2(-x^2{}_2 - e^{-x_1}\sin t) - e^{-x_1}\cos t \end{bmatrix}$$

식(14)를 식(10)에 넣어 정리하면 다음과 같은 비선형방정식을 구할 수 있다.

$$F_1 = x_1 - \frac{1}{3}\Delta T x_2 - C_1$$
$$F_2 = x_2 - \frac{1}{3}\Delta T(-x^2{}_2 - e^{-x_1}\sin t) - C_2 \quad (15)$$

식(15)의 방정식을 풀기위한 Jacobian은 다음과 같다.

$$\begin{bmatrix} \frac{\partial F_1}{\partial x_1} & \frac{\partial F_1}{\partial x_2} \\ \frac{\partial F_2}{\partial x_1} & \frac{\partial F_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 1 & \frac{1}{3}\Delta T \\ \frac{1}{3}\Delta T e^{-x_1}\sin t & 1 + \frac{2}{3}\Delta T x_2 \end{bmatrix} \quad (16)$$

2.3 불연속성이 포함된 경우

다음의 미분방정식을 고려해 보자. 이 경우는 불연속점이 포함된 함수이다.

$$\dot{x_1} = x^2{}_1(1 + t)/x_2$$
$$\dot{x_2} = -10[x_2 + (1 + t)x_1]x_2[x_2 - (1 + t)] + u(t) \quad (17)$$

여기서, $x_1(0) = 1, x_2(0) = -0.15$

$$u(t) = \begin{cases} 1 & (1 \le t \le 2) \\ 0 & (2 < t < 3) \\ -1 & (3 \le t \le 4) \\ 0 & (\text{otherwise}) \end{cases}$$

식(17)을 주어진 방법으로 풀면 다음과 같다.

$$F_1 = x_1 - \frac{1}{3}\Delta T(x^2{}_1(1 + t)/x_2) - C_1$$
$$F_2 = x_2 - \frac{1}{3}\Delta T(-10[x_2 + (1 + t)x_1]x_2[x_2 - (1 + t)] + u(t)) - C_2$$

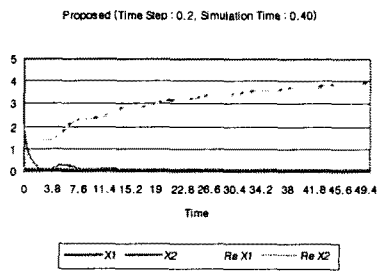
3. 시뮬레이션 결과

2절에서 수학적으로 전개한 미분방정식을 컴퓨터를 이용하여 결과를 구해본다.

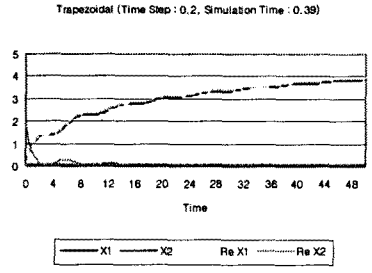
3.1 연속함수의 경우

그림 1에는 2.2절에서 보여진 미분방정식을 Runge-Kutta법과 Trapezoidal법과 제안된 방법으로 시뮬레이션을 한 그래프를 보여준다. 시뮬레이션 Time step은 0.2초로 비교적 크게 하였다. 세 가지 방법으로 구한 결과 그래프를 실제해의 그래프와 비교해 얼마나 정확한 결과가 나오는지를 보여준다. 시뮬레이션 시간은 0.4초와 0.39초로 비슷한 결과를 보여주고 있다. 결과는 Trapezoidal법이 약간의 오차를 보여주고 있으며, Runge-Kutta법과 제안된 방법을 거의 정확한 결과를 보여준다.

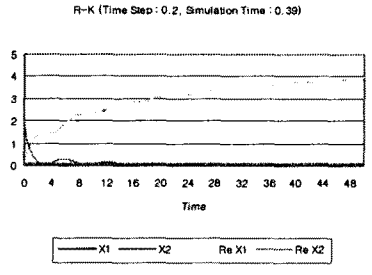
$$\dot{x_1} = x_2$$
$$\dot{x_2} = x^2{}_2 - e^{-x_1}\sin t \qquad x_1(0) = 0, x_2(0) = 2$$



(a) 제안된 방법



(b) Trapezoidal 방법



(c) Runge-Kutta 방법

그림 1. 시뮬레이션 결과(Time Step : 0.2초)

3.2 불연속 함수의 경우

표 1과 3은 2.3절의 미분방정식을 시뮬레이션한 결과를 보여준다. 표 1과 3은 Time step을 0.05초로 놓고 시

뮬레이션한 결과를 보여주고 있으며, 이 결과가 얼마나 정확한 결과인지를 비교하기 위해서 표 2와 4에서 Time step을 0.001초로 시뮬레이션한 결과를 보여주고 있다. 표 1과 3에서 오차는 Time step 0.001초로 시뮬레이션한 결과에서 Time step 0.05초로 시뮬레이션한 결과를 뺀 값으로 나타낸다. 표 1과 2에서는 x_2의 초기값을 -0.2로 놓고 시뮬레이션 결과이고, 표 3과 4에서는 x_2의 초기값을 0.15로 놓고 시뮬레이션한 결과를 보여준다.

표 1에서는 제안된 방법이 Trapezoidal법보다는 정확한 결과를 보여주고 Runge-Kutta법보다는 약간 부정확한 결과를 보여주고 있다. 반면 표 3에서 보여주는 것과 같이 severe한 경우에 있어서 Trapezoidal법은 2초를 지나면서 발산을 해버리는 반면, 제안된 방법은 Runge-Kutta법보다도 더 정확한 결과값을 보여주고 있다.

$$\dot{x_1} = x^2{}_1(1+t)/x_2$$

$$\dot{x_2} = -10[x_2 + (1+t)x_1]x_2[x_2 - (1+t)] + u(t)$$

여기서, $x_1(0) = 1, x_2(0) = -0.2$

제안된 방법			Trapezoidal법		Runge-Kutta법	
time	x_1	x_2	x_1	x_2	x_1	x_2
0	1	-0.2	1	-0.2	1	-0.2
0.05	0.831664	-0.29946	0.817516	-0.305194	0.828685	-0.306816
0.1	0.739284	-0.412421	0.725774	-0.42283	0.738455	-0.430006
...	...	...	...	...	...	...
0.95	0.298807	-0.600362	0.295418	-0.593272	0.301927	-0.60588
1	0.284557	-0.574108	0.281211	-0.561962	0.287616	-0.584241
오차	0.003115	-0.018278	0.006461	-0.030424	0.000056	-0.008145
1.05	0.270443	-0.535701	0.267155	-0.521047	0.27339	-0.539773
...	...	...	...	...	...	...
2	0.084585	-0.154258	0.079489	-0.123868	0.084059	-0.145145
오차	-0.000245	0.007684	0.004851	-0.022706	0.000281	-0.001429
2.05	0.077816	-0.145989	0.072327	-0.119535	0.077378	-0.158943
...	...	...	...	...	...	...
3	0.03034	-0.151344	0.027468	-0.145121	0.030566	-0.145008
오차	0.000697	0.12345	0.003569	0.006122	0.000471	0.006009
3.05	0.02928	-0.1824	0.026544	-0.176858	0.029455	-0.179905
...	...	...	...	...	...	...
4	0.018128	-0.193287	0.016874	-0.189029	0.018201	-0.193448
오차	0.000241	-0.00074	0.001495	-0.004998	0.000168	-0.000579
4.05	0.017699	-0.178488	0.016483	-0.169341	0.017745	-0.159953
...	...	...	...	...	...	...
5	0.008408	-0.060962	0.00785	-0.057071	0.008289	-0.059751
오차	-0.000076	0.000926	0.000482	-0.002965	0.000043	-0.000285

표 1. 시뮬레이션 결과($x_1(0) = 1,\ x_2(0) = -0.2$)

Time Step : 0.001초		
time	x_1	x_2
1	0.287672	-0.592386
2	0.08434	-0.146574
3	0.031037	-0.138999
4	0.018369	-0.194027
5	0.008332	-0.060036

표 2. 시뮬레이션 결과(Time Step : 0.001초)

$$\dot{x_1} = x^2{}_1(1+t)/x_2$$

$$\dot{x_2} = -10[x_2 + (1+t)x_1]x_2[x_2 - (1+t)] + u(t)$$

여기서, $x_1(0) = 1, x_2(0) = -0.15$

제안된 방법			Runge-Kutta법		Trapezoidal법	
time	x_1	x_2	x_1	x_2	x_1	x_2
0	1	-0.15	1	-0.15	1	-0.15
0.05	0.790712	-0.225702	0.783883	-0.23001	0.766002	-0.228756
...	...	...	...	...	...	...
0.95	0.265103	-0.534709	0.267203	-0.538383	0.258384	-0.521265
1	0.252495	-0.509665	0.254588	-0.518393	0.245978	-0.490968
오차	0.00227	-0.017105	0.000177	-0.008377	0.008787	-0.035802
1.05	0.239914	-0.470504	0.241957	-0.473508	0.233595	-0.449526
...	...	...	...	...	...	...
2	0.057118	-0.035967	0.047229	-0.010812	-0.008377	0.049461
오차	-0.006536	0.020968	0.003353	-0.004187	0.058959	-0.06446
2.05	0.038064	-0.008832	0.031348	-0.002848	-0.008206	0.07846
...	...	...	...	...	...	...
3	0.003738	-0.031622	0.000699	-0.011464	-	-
오차	0.002534	0.002988	0.005573	-0.01717	-	-
3.05	0.00369	-0.076605	0.000695	-0.058691	-	-
...	...	...	...	...	-	-
4	0.003329	-0.148806	0.00068	-0.142098	-	-
오차	0.001874	-0.005011	0.004523	-0.011719	-	-
4.05	0.00331	-0.134651	0.000679	-0.108895	-	-
...	...	...	...	...	-	-
5	0.0023	-0.025338	0.000607	-0.017959	-	-
오차	0.00087	-0.003956	0.002563	-0.011335	-	-

표 3. 시뮬레이션 결과($x_1(0) = 1,\ x_2(0) = -0.15$)

Time Step : 0.001초		
time	x_1	x_2
1	0.254765	-0.52677
2	0.050582	-0.014999
3	0.006272	-0.028634
4	0.005203	-0.153817
5	0.00317	-0.029294

표 4. 시뮬레이션 결과(Time Step : 0.001초)

4. 결 론

이 논문에서는 미분방정식의 해를 구하는 새로운 알고리즘을 제시하고, 그 결과를 Runge-Kutta법과 Trapezoidal법의 결과와 비교하였다. 시뮬레이션 시간에 있어서는 세 가지 방법이 거의 같은 시간을 소모하였다. 일반적인 연속함수의 경우에서는 Runge-Kutta법이 제일 정확한 결과값을 보여주었으며 제안된 방법도 Runge-Kutta법에 거의 비슷한 성능을 보였고 Trapezoidal법은 두 방법보다는 큰 오차를 보였다. 불연속점이 있는 경우와 같이 severe한 경우에서는 Trapezodal법은 발산을 하였고, 제안된 방법이 Runge-Kutta법보다도 더 정확한 결과를 보여주었다.

[참 고 문 헌]

[1] Eugene V. Solodovnik, Geroge J. Cokkinides, A. P. Sakis Meliopoulos, "On Stability of Implicit Numerical Methods in Nonlinear Dynamical Systems Simulation", System Theory, 1998. Proceedings of the Thirtieth Southeastern Symposium on 8-10 Mar. 1998, page(s) : 27-31

[2] A.C. Logovski, "Methods for Solving of Differential Equations in Neural Basis", Neuroinfformationcs and Neurocomputers, 1992, RNNS/IEEE Symposium on, 7-10 Oct. 1992, page(s) : 919-927 vol. 2

[3] Rafic Braham, "Numerical Analysis of Neural Networks", Neural Networks, 1989, IJCNN., International Joint Conference on, 18-22 Jun. 1989, page(s) : 425-428 vol. 1

[4] Chiu H. Choi, "A Survey of Numerical Methods for Solving Matrix Riccati Differential Equations", Southeastcon '90, Proceedings., IEEE, 1-4 Apr. 1990, page(s) : 696-700 vol. 2