第39卷第2期
2011年2月

同济大学学报(自然科学版)
JOURNAL OF TONGJI UNIVERSITY(NATURAL SCIENCE)

Vol.39 No.2
Feb. 2011

文章编号：0253-374X(2011)02-0179-08

DOI:10.3969/j.issn.0253-374x.2011.02.005

中心-偏心差分法

李常青[1,2]，楼梦麟[1]

(1.同济大学 土木工程防灾国家重点实验室，上海 200092；2.中南大学 土木建筑学院，湖南 长沙 410075)

摘要：提出了构造差分显式位移动力算法的通式，并得到了三步显式位移算法.此法对加速度采用中心差分近似，但对速度采用三点偏心差近似，因此也称为中心-偏心差分法.此法可以认为是对中心差分法的改进，克服了中心差分法在计算阻尼矩阵为非对角阵时退化为隐式算法的缺点.中心-偏心差分法的算法精度为二阶.对此算法的稳定性进行了分析.分析表明，与同类显式算法相比，本方法具有时间和空间两方面的算法优势.

关键词：显式算法；中心差分法；偏心差分

中图分类号：O 321　　**文献标识码**：A

Central-eccentric Difference Method

LI Changqing[1,2], *LOU Menglin*[1]

(1. State Key Laboratory of Disaster Reduction in Civil Engineering, Tongji University, Shanghai 200092, China; 2. School of Civil Engneering and Architecture, Central South University, Changsha 410075, China)

Abstract: A kind of general explicit displacement algorithm is derived and three-step explicit displacement method with second-order precision is obtained. The acceleration is approximated by central difference and the velocity is approximated by three-point eccentric difference. Therefore, this method is also called central-eccentric difference method, which can be looked as an improved form of central difference method because it remians as an explicit method when the damp matrix is not diagonal. The stabilization of this method is analyzed and results show that this method is superior in costs of calculation time and storage space over the other similar explicit methods.

Key words: explicit method; central difference method; eccentric

结构动力计算采用的逐步积分方法有显式算法和隐式算法之分.隐式算法需要求解耦联方程组，当结构自由度数目很大，譬如上百万个时，求解这一方程组的工作量非常大.而显式算法不需要求解方程组，而计算精度控制的要求往往比稳定性条件要求更高.通常情况下需要考虑体系非线性时，采用条件稳定的显式格式求解动力反应是非常有利的.因此，显式积分方法在许多工程领域内不断受到人们的关注.中心差分法作为显式算法时只能处理阻尼矩阵为对角矩阵时的情况，当阻尼矩阵为非对角矩阵时中心差分法退化为隐式算法.而用单边差分方法来处理速度项时，虽然对阻尼矩阵为非对角阵时显式算法仍然有效，但是算法的精度只有一阶.对此，李小军等[1-2]采用中心差分法及Newmark方法的基本假定相结合的方法给出了求解动力方程的一种显式差分格式.杜修力等[3]提出了一种阻尼弹性结构动力计算的显式差分法.后来王进廷等[4]提出了一种新的二阶精度的显式算法.张晓志等[5]利用荷载和荷载的时变化率推导了三阶的显式方程，但是用到了荷载向量的一阶导数，因此实际能达到的计算精度受高阶数值微分精度的限制. 陈学良等[6]发展了一种直接以地震加速度作为输入的显式算法，该算法以中心差分和Newmark-β法结合，并通过平衡方程约简得到.这些算法在构造过程中，至少位移和速度项都是必需的计算项.本文探讨只需要位移项参与的显式算法，以提高计算效率，节省计算空间.

1　差分近似导数的公式构造和精度分析

本文算法基于用差分近似导数来得到，因此首

收稿日期：2009-10-22
基金项目：国家自然科学基金重点项目(90915011)；上海科委基础研究重点项目(07JC14051)
第一作者：李常青(1978—)，男，博士生，主要研究方向为复杂结构的地震反应分析和动力方程求解算法.E-mail:lcq_stu@126.com
通讯作者：楼梦麟(1947—)，男，教授，博士生导师，工学博士，主要研究方向为工程结构抗震与震动控制、土-结构动力相互作用和复杂结构动力计算理论.E-mail:lml@tongji.edu.cn

先涉及到的便是差分近似导数的公式构造和精度分析.

对函数 $x(t)$在点 t_0 泰勒展开,可以得到一个多项式用来描叙此函数的局部变化信息.反之可利用已知 t_0 相邻点的 x 值的信息,通过构造不同的格式来得到 t_0 点的 x 的各阶导数不同精度的近似值.通常的做法是利用 t_0 点及其附近两点 t_{-1}和 t_1 的 x 的值来构造一阶和二阶导数的差分格式,只有构造三阶及三阶以上的导数的差分近似格式时才利用 t_0 点附近更多的点的 x 值.本文在构造一阶和二阶导数的差分格式时采用 t_0 点更多的附近点的 x 值.

推导过程如下:假定 $x(t)$有直到$(n+1)$阶的导数,即 $x(t)$为$(n+1)$阶可微函数.在 t_0 附近进行泰勒展开,则具有皮亚诺余项的公式为

$$x_i = x_0 + (i\Delta t)\dot{x}_0 + \frac{1}{2}(i\Delta t)^2\ddot{x}_0 + \cdots + \frac{1}{m!}(i\Delta t)^m(x_0)^{(m)} + \cdots + \frac{1}{n!}(i\Delta t)^n(x_0)^{(n)} + o(\Delta t)^n \quad i = -j, -j+1, \cdots, k \tag{1}$$

式中:Δt 为时间步距;$x_i = x(t)\Big|_{t=t_i}$,$(x_0)^{(k)} = \dfrac{d^k x}{dt^k}\Big|_{x=x_0}$.这里利用了 t_0 点附近的前面 j 个点和后面 k 个点的 x 的值的信息,其中 j,k 为整数.式(1)乘以系数 l_i,得到式(2),其中 l_i 为待确定量,$i=-j,-j+1,\cdots,k$ 且 $i\neq 0$.

$$l_i x_i = l_i x_0 + (i\Delta t)l_i\dot{x}_0 + \frac{1}{2}(i\Delta t)^2 l_i\ddot{x}_0 + \cdots + \frac{1}{m!}(i\Delta t)^m l_i(x_0)^{(m)} + \cdots + \frac{1}{n!}(i\Delta t)^n l_i(x_0)^{(n)} + o(\Delta t)^n l_i \quad i = -j, -j+1, \cdots, k, i\neq 0 \tag{2}$$

式(2)包含的是 $j+k$ 个方程,将这些方程累加,得

$$\sum_{i=-j}^{k} l_i x_i = \sum_{i=-j}^{k}\Big[l_i x_0 + (i\Delta t)l_i\dot{x}_0 + l_i\frac{1}{2}(i\Delta t)^2\cdot l_i\ddot{x}_0 + \cdots + \frac{1}{m!}(i\Delta t)^m l_i(x_0)^{(m)} + \cdots + \frac{1}{n!}(i\Delta t)^n l_i(x_0)^{(n)} + o(\Delta t)^n l_i\Big] \tag{3}$$

式(3)改写为

$$\sum_{i=-j}^{k} l_i(x_i - x_0) = \sum_{i=-j}^{k}\Big[(i\Delta t)l_i\dot{x}_0 + \frac{1}{2}(i\Delta t)^2\cdot l_i\ddot{x}_0 + \frac{1}{m!}(i\Delta t)^m l_i(x_0)^{(m)} + \cdots + \frac{1}{n!}(i\Delta t)^n l_i(x_0)^{(n)} + o(\Delta t)^n l_i\Big] \tag{4}$$

式(4)显示了 t_0, t_0 前 j 点和 t_0 后 k 点的 x 的值与 t_0 点处 x 的各阶导数值之间的关系.其中,可变的参数有 $j+k$ 个.因此,为了得到关于 t_0 点的 x 的各阶导数的高精度的近似,可以按照所求的导数的阶以及精度要求的不同来选择 j 和 k 的取值,最后得到用 t_0 附近 $j+k$ 个不同点的 x 的值以及 x_0 本身来表示的近似差分表达式.为了区分不同情况下的公式推导,式(4)中的 j,k,l_i 在建立速度的差分格式时,都加上 v,以表示和速度相关,变为 j_v,k_v,l_{vi};而在建立加速度的差分格式时,都加上 a,以表示和加速度相关,变为 j_a,k_a,l_{ai}.具体计算过程如下.

1.1 速度的差分格式

式(4)中的 j,k,l_i 分别用j_v,k_v,l_{vi}代替,得

$$\sum_{i=-j_v}^{k_v} l_{vi}(x_i - x_0)/\Delta t\sum_{i=-j_v}^{k_v}(il_{vi}) = \dot{x}_0 + \sum_{i=-j_v}^{k_v}\Big[\frac{1}{2}(i\Delta t)^2 l_{vi}\ddot{x}_0 + \cdots + \frac{1}{m!}(i\Delta t)^m l_{vi}(x_0)^{(m)}\Big]/\Delta t\sum_{i=-j_v}^{k_v}(il_{vi}) + \sum_{i=-j_v}^{k_v}\Big[\frac{1}{(m+1)!}(i\Delta t)^{m+1} l_{vi}(x_0)^{(m+1)} + \cdots + \frac{1}{n!}(i\Delta t)^n l_{vi}(x_0)^{(n)} + o(\Delta t)^n l_{vi}\Big]/\Delta t\sum_{i=-j_v}^{k_v}(il_{vi}) \tag{5}$$

式(5)可以改为

$$\dot{x}_0 = \sum_{i=-j_v}^{k_v} l_{vi}(x_i - x_0)/\Delta t\sum_{i=-j_v}^{k_v}(il_{vi}) - \sum_{i=-j_v}^{k_v}\Big[\frac{1}{2}(i\Delta t)^2 l_{vi}\ddot{x}_0 + \cdots + \frac{1}{m!}(i\Delta t)^m l_{vi}(x_0)^{(m)}\Big]/\Delta t\sum_{i=-j_v}^{k_v}(il_{vi}) - \sum_{i=-j_v}^{k_v}\Big[\frac{1}{(m+1)!}(i\Delta t)^{m+1} l_{vi}(x_0)^{(m+1)} + \cdots + \frac{1}{n!}(i\Delta t)^n l_{vi}(x_0)^{(n)} + o(\Delta t)^n l_{vi}\Big]/\Delta t\sum_{i=-j_v}^{k_v}(il_{vi}) \tag{6}$$

式(5)的左边,也就是得到的差分近似表达式,用它来近似表示$\dot{x}_0$.为了得到 m 阶精度,需要使式(5)中 $\ddot{x}_0,\dddot{x}_0\cdots,x_0^{(m)}$ 项的系数为零.此时式(5) 为

$$\Big[\sum_{i=-j_v}^{k_v} l_{vi}x_i - \sum_{i=-j_v}^{k_v} l_{vi}x_0\Big]/\Delta t\sum_{i=-j_v}^{k_v}(il_{vi}) =$$

$$\dot{x}_0 + \left\{ \sum_{i=-j_v}^{k_v} \frac{1}{(m+1)!}(i)^{m+1} l_{vi} (x_0)^{(m+1)} \Delta t^m + \cdots + \sum_{i=-j_v}^{k_v} \frac{1}{n!}(i)^n l_{vi} (x_0)^{(n)} \Delta t^{n-1} + o(\Delta t)^{n-1} l_{vi} \right\} \Big/ \sum_{i=-j_v}^{k_v} (i l_{vi}) = \dot{x}_0 + M \Delta t^m \tag{7}$$

式中,M 为与 Δt 无关的有界函数.式(7)是构造 $\dot{x}_0$ 的具有 m 阶精度的差分近似表达式的依据.这样,$\dot{x}_0$ 的具有 m 阶精度的近似公式可写为

$$\dot{x}_0 \approx \Big[\sum_{i=-j_v}^{k_v} l_{vi} x_i - \sum_{i=-j_v}^{k_v} l_{vi} x_0 \Big] / \Delta t \sum_{i=-j_v}^{k_v} (i l_{vi}) \tag{8}$$

要使 $\ddot{x}_0, \dddot{x}_0, \cdots, x_0^{(m)}$ 的系数为零,需满足如下条件:

$$\sum_{i=-j_v}^{k_v} \frac{(i\Delta t)^2}{2} l_{vi} = 0, \sum_{i=-j_v}^{k_v} \frac{(i\Delta t)^3}{6} l_{vi} = 0, \cdots, \sum_{i=-j_v}^{k_v} \frac{(i\Delta t)^m}{m!} l_{vi} = 0$$

此为关于 $l_{vi}, i = -j_v, -j_v+1, \cdots, k_v$ 且 $i \neq 0$ 的线性齐次方程组,可以化简为

$$\begin{cases} \sum_{i=-j_v}^{k_v} (i)^2 l_{vi} = 0 \\ \sum_{i=-j_v}^{k_v} (i)^3 l_{vi} = 0 \\ \vdots \\ \sum_{i=-j_v}^{k_v} (i)^m l_{vi} = 0 \end{cases} \tag{9}$$

写成矩阵形式为

$$\begin{bmatrix} (-j_v)^2 & (-j_v+1)^2 & \cdots & (-1)^2 & 1^2 & \cdots & (k_v)^2 \\ (-j_v)^3 & (-j_v+1)^3 & & (-1)^3 & 1^3 & & (k_v)^3 \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ (-j_v)^m & (-j_v+1)^m & \cdots & (-1)^m & 1^m & \cdots & (k_v)^m \end{bmatrix} \begin{Bmatrix} l_{v(-j_v)} \\ l_{v(-j_v+1)} \\ \vdots \\ l_{v(-1)} \\ l_{v(1)} \\ \vdots \\ l_{v(k_v)} \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \end{Bmatrix} \tag{10}$$

式(10)为 $m-1$ 个方程,求解 $j_v + k_v$ 个待求量.当 $m-1 = j_v + k_v$ 时,也就是方程的数目与未知量的数量相等时可用数学归纳法证明,方程组的系数矩阵为满秩的方阵只有零解.当 $m-1 = j_v + k_v - 1$,即 $m = j_v + k_v$,也即方程的数量比未知量的数目少 1 个,此时系数矩阵为行满秩矩阵,方程组(10)有非零解,基础解系的自由度为 1,可以得到个各未知量之间一个固定的比值关系,将这种比值关系代入式(8),可得到差分表达式.总结为,用 t_0 附近的 $j_v + k_v$ 个点的 x 的值可以得到速度 $\dot{x}_0$ 的 $j_v + k_v$ 阶精度的近似差分表达式.

取 $j_v = 2, k_v = 0$,可得到 3 点表示的 $j_v + k_v = 2$ 的二阶精度的速度表达式.此时式(10)为

$$4 l_{v(-2)} + l_{v(-1)} = 0 \tag{11}$$

式(11)的解为

$$\begin{cases} l_{v(-1)} = 4c_1 \\ l_{v(-2)} = -c_1 \end{cases} \tag{12}$$

式中,c_1 为任意非零常数.

将式(12)代入式(8),可得到 3 点表示的二阶精度的速度近似表达式,其中 c_1 在代入过程中被消掉.

$$\dot{x}_0 \approx \Big[\sum_{i=-2}^{0} l_{vi} x_i - \sum_{i=-2}^{0} l_{vi} x_0 \Big] / \Delta t \sum_{i=-2}^{0} (i l_{vi}) = (3x_0 - 4x_{-1} + x_{-2}) / 2\Delta t \tag{13}$$

取 $j_v = 2, k_v = 1$,此时式(10)为

$$\begin{cases} 4 l_{v(-2)} + l_{v(-1)} + l_{v(1)} = 0 \\ -8 l_{v(-2)} - l_{v(-1)} + l_{v(1)} = 0 \end{cases} \tag{14}$$

求解式(14)可得到

$$l_{v(1)} = 2c_2, l_{v(-1)} = -6c_2, l_{v(-2)} = c_2 \tag{15}$$

式中,c_2 为任意非零常数.

将式(15)代入式(8)可得 4 点表示的三阶精度的速度表达式,其中 c_2 在代入过程中被消掉.

$$\dot{x}_0 = \Big[\sum_{i=-2}^{1} l_{vi} x_i - \sum_{i=-2}^{1} l_{vi} x_0 \Big] / \sum_{i=-2}^{1} (i\Delta t) l_{vi} = (-6x_{-1} + x_{-2} + 2x_1 + 3x_0) / 6\Delta t \tag{16}$$

1.2　加速度的差分格式

按照相同的步骤可以证明,利用 t_0 附近 $j_a + k_a$ 个点的 x 值,可以得到加速度 $\ddot{x}_0$ 的最大 $j_a + k_a - 1$ 阶精度的近似差分表达式.

式(4)中 j,k,l_i 分别用 j_a,k_a,l_{ai} 来代替,得

$$\ddot{x}_0 = \sum_{i=-j_a}^{k_a} l_{ai}(x_i - x_0)/\sum_{i=-j_a}^{k_a}\frac{1}{2}(i\Delta t)^2 l_{ai} - \sum_{i=-j_a}^{k_a}\left[(i\Delta t)l_{ai}\dot{x}_0 + \frac{1}{3!}(i\Delta t)^3 l_{ai}(\dddot{x}_0) + \cdots + \frac{1}{m!}(i\Delta t)^m l_{ai}(x_0)^{(m)} + \cdots + \frac{1}{n!}(i\Delta t)^n \cdot l_{ai}(x_0)^{(n)} + o(\Delta t)^n l_{ai}\right]/\sum_{i=-j_a}^{k_a}\frac{1}{2}(i\Delta t)^2 l_{ai} \tag{17}$$

使式(17)中 $\dot{x}_0,\dddot{x}_0\cdots,x_0^{(m)}$ 的系数为零,可得关于 $\ddot{x}_0$ 的 $j_a + k_a - 1$ 阶的精度的近似表达式

$$\ddot{x}_0 = \sum_{i=-j_a}^{k_a} l_{ai}(x_i - x_0)/\sum_{i=-j_a}^{k_a}\frac{1}{2}(i\Delta t)^2 l_{ai} \tag{18}$$

$\dot{x}_0,\dddot{x}_0\cdots,x_0^{(m)}$ 的系数为零,即要求方程组(19)成立

$$\begin{cases}\sum\limits_{i=-j_a}^{k_a} i l_{ai} = 0\\ \sum\limits_{i=-j_a}^{k_a} i^3 l_{ai} = 0\\ \vdots\\ \sum\limits_{i=-j_a}^{k_a} i^m l_{ai} = 0\end{cases} \tag{19}$$

取 $j_a=2,k_a=1$,方程组(19)可写为

$$\begin{cases}-2l_{a(-2)} - 1l_{a(-1)} + l_{a(1)} = 0\\ -8l_{a(-2)} - 1l_{a(-1)} + l_{a(1)} = 0\end{cases} \tag{20}$$

解方程组(20),可得到

$$\begin{cases}l_{a(1)} = -c_3\\ l_{a(-1)} = c_3\\ l_{a(-2)} = 0\end{cases} \tag{21}$$

式中,c_3 为任意非零常数.将式(21)代入式(18),可以得到

$$\ddot{x}_0 = \sum_{i=-1}^{1} l_{ai}(x_i - x_0)/\sum_{i=-1}^{1}\frac{1}{2}(i\Delta t)^2 l_{ai} = (x_{-1} + x_1 - 2x_0)/\Delta t^2 \tag{22}$$

这就是中心差分公式.

取 $j_a=2,k_a=2$,方程组(19)可写为

$$\begin{cases}-2l_{a(-2)} - l_{a(-1)} + l_{a(1)} + 2l_{a(2)} = 0\\ -8l_{a(-2)} - l_{a(-1)} + l_{a(1)} + 8l_{a(2)} = 0\\ 16l_{a(-2)} + l_{a(-1)} + l_{a(1)} + 16l_{a(2)} = 0\end{cases} \tag{23}$$

解方程组(23),可得到

$$\begin{cases}l_{a(-1)} = l_{a(1)} = 16c_4\\ l_{a(-2)} = l_{a(2)} = -1c_4\end{cases} \tag{24}$$

式中,c_4 为任意非零常数.将式(24)代入式(18)得

$$\ddot{x}_0 = (-x_{-2} + 16x_{-1} + 16x_1 - x_2 - 30x_0)/12\Delta t^2 \tag{25}$$

这是5点表示的加速度的差分近似表达式,有三阶精度.

上面的推导可以总结为:利用 t_0 附近 $j_v + k_v$ 个点的 x 值可以得到速度 $\dot{x}_0$ 的最大 $j_v + k_v$ 阶精度的差分近似表达式.利用 t_0 附近 $j_a + k_a$ 个点的 x 值可以得到加速度的最大 $j_a + k_a - 1$ 阶精度的近似差分表达式.因此,差分格式中的点增加1个,可以提高差分近似导数一阶精度.

2 显式算法推导

通常结构动力学方程为

$$\boldsymbol{M}\ddot{\boldsymbol{x}} + \boldsymbol{C}\dot{\boldsymbol{x}} + \boldsymbol{K}\boldsymbol{x} = \boldsymbol{F}(t) \tag{26}$$

式中:$\boldsymbol{M},\boldsymbol{C},\boldsymbol{K}$ 分别为结构的质量、阻尼、刚度矩阵;$\ddot{\boldsymbol{x}},\dot{\boldsymbol{x}},\boldsymbol{x}$ 分别为加速度、速度和位移向量;$\boldsymbol{F}(t)$ 为荷载向量.

在各时间点,有平衡方程,这里指定在 t_0 点有动力平衡方程

$$\boldsymbol{M}\ddot{\boldsymbol{x}}_0 + \boldsymbol{C}\dot{\boldsymbol{x}}_0 + \boldsymbol{K}\boldsymbol{x}_0 = \boldsymbol{F}(t_0) \tag{27}$$

式中,$\ddot{\boldsymbol{x}}_0 = \ddot{\boldsymbol{x}}\mid_{t=t_0}$,$\dot{\boldsymbol{x}}_0 = \dot{\boldsymbol{x}}\mid_{t=t_0}$,$x_0 = x\mid_{t=t_0}$.

用差分近似表示速度涉及到 t_0 点前 j_v 个点、后 k_v 个点,用差分近似表示加速度用到 t_0 点前 j_a 个点、后 k_a 个点.因此显式算法构造格式的取点范围就用到了 t_0 点前 p 个点、后 q 个点,其中,p 取 j_v 和 j_a 的最大值,q 取 k_v 和 k_a 的最大值,见图1.本文推导的多步法,就是利用 t_0 点前 p 个点、后 $q-1$ 个点的位移以及 t_0 本身的位移得到第 q 点的位移.

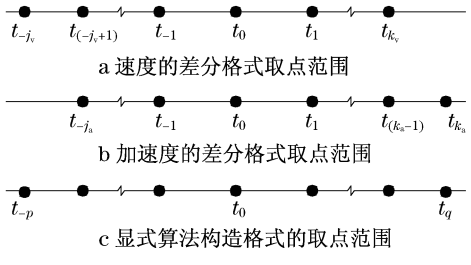


图1 显式算法构造格式的取点范围示意图

Fig.1 Time points used in the multi-step explicit displacement method

式(8)可以改写为

$$\dot{\boldsymbol{x}}_0 = \left(\sum_{i=-j_v}^{k_v} r_{vi}\boldsymbol{x}_i\right) \tag{28}$$

式中:$r_{vi} = l_{vi}/\sum_{i=-j_v}^{k_v}(i\Delta t)\,l_{vi}$, $i = -j_v, -j_v+1, \cdots, k_v$ 且 $i\neq 0$, $r_0 = \sum_{i=-j_v}^{k_v} l_{vi}/\sum_{i=-j_v}^{k_v}(i\Delta t)\,l_{vi}$.

式(18)可以改写为

$$\ddot{\boldsymbol{x}}_0 = \left(\sum_{i=-j_a}^{k_a} r_{ai}\boldsymbol{x}_i\right) \tag{29}$$

式中:$r_{ai} = l_{ai}/\sum_{i=-j_a}^{k_a}\frac{1}{2}(i\Delta t)^2 l_{ai}$, $i = -j_a, -j_a+1, \cdots, k_a$ 且 $i\neq 0$, $r_{a(0)} = -\sum_{i=-j_a}^{k_a} l_{ai}/\sum_{i=-j_a}^{k_a}\frac{1}{2}\cdot(i\Delta t)^2 l_{ai}$.

将式(28)~(29)代入方程(26),可得到

$$\boldsymbol{M}\left(\sum_{i=-j_a}^{k_a} r_{ai}\boldsymbol{x}_i\right)+\boldsymbol{C}\left(\sum_{i=-j_v}^{k_v} r_{vi}\boldsymbol{x}_i\right)+\boldsymbol{K}\boldsymbol{x}_0 = \boldsymbol{F}_0 \tag{30}$$

式(30)改写为

$$\boldsymbol{M}r_{a(k_a)}\boldsymbol{x}_{k_a} + \boldsymbol{M}\left(\sum_{i=-j_a}^{k_a-1} r_{ai}\boldsymbol{x}_i\right)+ \boldsymbol{C}\left(\sum_{i=-j_v}^{k_v} r_{vi}\boldsymbol{x}_i\right)+\boldsymbol{K}\boldsymbol{x}_0 = \boldsymbol{F}_0 \tag{31}$$

为了得到显式算法,必须使所要求的计算步的位移值不能出现在 $\boldsymbol{C}$ 和 $\boldsymbol{K}$ 相乘的式子中(当阻尼矩阵对角时,可以出现在矩阵 $\boldsymbol{C}$ 中),但必须出现在 $\boldsymbol{M}$ 相乘的式子中,因此要求 $r_{a(k_a)}\neq 0$, $r_{a(k_a)} > r_{v(k_v)}$,也即通过 t_0 点的动力平衡来建立多步显式位移法必须满足的条件,为

$$\begin{cases} k_a \geqslant 1, k_a \geqslant k_v & \boldsymbol{C}\text{为对角矩阵} \\ k_a \geqslant 1, k_a > k_v & \boldsymbol{C}\text{为任意矩阵} \end{cases} \tag{32}$$

式(31)可以改写为

$$\boldsymbol{x}_{k_a} = \frac{1}{r_{a(k_a)}}\boldsymbol{M}^{-1}\boldsymbol{F}_0 - \frac{1}{r_{a(k_a)}}\left(\sum_{i=-j_a}^{k_a-1} r_{ai}\boldsymbol{x}_i\right) - \frac{1}{r_{a(k_a)}}\boldsymbol{M}^{-1}\boldsymbol{C}\left(\sum_{i=-j_v}^{k_v} r_{vi}\boldsymbol{x}_i\right) - \frac{1}{r_{a(k_a)}}\boldsymbol{M}^{-1}\boldsymbol{K}\boldsymbol{x}_0 \tag{33}$$

这就得到了利用时间点 t_0 的动力平衡得到的多步显式位移法. 中心差分法对应的是 $j_v = k_v = j_a = k_a = 1$.

3　二阶精度三步显式位移算法

对速度用差分近似时,取 $j_v = 2$, $k_v = 0$,对加速度用差分近似时取 $j_a = 2$, $k_a = 1$ 来近似. 也即将式(13)、式(22)代入式(26),可得到

$$\boldsymbol{M}(\boldsymbol{x}_{i+1} + \boldsymbol{x}_{i-1} - 2\boldsymbol{x}_i)/\Delta t^2 + \boldsymbol{C}(3\boldsymbol{x}_i - 4\boldsymbol{x}_{i-1} + \boldsymbol{x}_{i-2})/(2\Delta t) + \boldsymbol{K}\boldsymbol{x}_i = \boldsymbol{F}_i \tag{34}$$

$$\boldsymbol{x}_{i+1} = \frac{1}{2}(-3\boldsymbol{M}^{-1}\boldsymbol{C}\Delta t - 2\boldsymbol{M}^{-1}\boldsymbol{K}\Delta t^2 + 4\boldsymbol{I})\boldsymbol{x}_i + (2\boldsymbol{M}^{-1}\boldsymbol{C}\Delta t - \boldsymbol{I})\boldsymbol{x}_{i-1} - \frac{1}{2}\boldsymbol{M}^{-1}\boldsymbol{C}\Delta t\boldsymbol{x}_{i-2} + \Delta t^2\boldsymbol{M}^{-1}\boldsymbol{F}_i \tag{35}$$

式中:$\boldsymbol{I}$ 为单位矩阵. 式(35)即二阶精度的三步显式位移法.

如果对速度用差分近似时,取 $j_v = 2$, $k_v = 1$,对加速度用差分近似时取 $j_a = 2$, $k_a = 2$ 来近似. 也就是将式(16)、式(25)代入式(26),可以得到

$$\boldsymbol{x}_2 = (-\boldsymbol{x}_{-2} + 16\boldsymbol{x}_{-1} - 30\boldsymbol{x}_0 + 16\boldsymbol{x}_1) + 2\Delta t\boldsymbol{M}^{-1}\boldsymbol{C}(\boldsymbol{x}_{-2} - 6\boldsymbol{x}_{-1} + 3\boldsymbol{x}_0 + 2\boldsymbol{x}_1) + 12\Delta t^2\boldsymbol{M}^{-1}\boldsymbol{K}\boldsymbol{x}_0 - 12\Delta t^2\boldsymbol{M}^{-1}\boldsymbol{F}_0 \tag{36}$$

这是四步三阶精度的显式计算公式,但分析表明式(36)不稳定,不能采用. 因此本文只对三步二阶精度的显式位移法式(35)进行更深入的论述.

下面对此算法的精度进行验证. 已知式(13)关于速度的精度为二阶,式(22)关于加速度的精度为二阶,下面考察式(35)关于位移的精度.

式(35)在单自由度时写成准确的形式为

$$x_{i+1} = \frac{1}{2}(-6\zeta\Omega - 2\Omega^2 + 4)x_i + (4\zeta\Omega - 1)x_{i-1} - \zeta\Omega x_{i-2} + \Delta t^2\frac{F_i}{m} + \sigma_x \tag{37}$$

式中:$\zeta = \frac{c}{2m\omega}$, c 为阻尼系数, m 为质量, $\omega = \sqrt{\frac{k}{m}}$, k 为弹性系数; $\Omega = \omega\Delta t$; σ_x 为由计算格式(35)引入的位移误差.

将 x_{i-1}, x_{i-2} 作泰勒展开,得

$$\begin{cases} x_{i-1} = x_i - \dot{x}_i\Delta t + \frac{1}{2}\ddot{x}_i\Delta t^2 - \frac{1}{3!}\dddot{x}_i\Delta t^3 + \frac{1}{4!}x_i^{(4)}\Delta t^4 + \cdots \\ x_{i-2} = x_i - 2\dot{x}_i\Delta t + 2\ddot{x}_i\Delta t^2 - \frac{4}{3}\dddot{x}_i\Delta t^3 + \frac{4}{3}x_i^{(4)}\Delta t^4 + \cdots \end{cases} \tag{38}$$

将式(38)代入式(37),并利用动力方程下式:

$$mx_i + c\dot{x}_i + kx_i = f_i \tag{39}$$

可由式(38)得到

$$\sigma_x = \Delta t^3\left(\frac{2}{3}\Omega\zeta\dddot{x}_0 - \frac{1}{360}(30+180\Omega\zeta)\cdot x_0^{(4)}\Delta t + \cdots\right) = O(\Delta t^3) \tag{40}$$

本法关于位移为三阶精度.综合考察,本算法为二阶精度.

4 稳定性

一个算法的稳定性分析一般是求解一个单自由度的运动方程.对于单自由度系统,重写式(35),可以得到如下的单自由度系统计算的传递格式:

$$\begin{Bmatrix} x_1 \\ x_0 \\ x_{-1} \end{Bmatrix} = \begin{bmatrix} -\left(\frac{1.5c\Delta t}{m} - 2 + \frac{k\Delta t^2}{m}\right) & -\left(1-\frac{2c\Delta t}{m}\right) & -\frac{c\Delta t}{2m} \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot \begin{Bmatrix} x_0 \\ x_{-1} \\ x_{-2} \end{Bmatrix} + \begin{Bmatrix} f_0\Delta t^2/m \\ 0 \\ 0 \end{Bmatrix} \tag{41}$$

将 $c=2\xi\omega m$, $k=m\omega^2$ 代入式(41),可以得到更常用的表达形式.

$$\begin{Bmatrix} x_1 \\ x_0 \\ x_{-1} \end{Bmatrix} = \begin{bmatrix} -3\xi\omega\Delta t + 2 - (\omega\Delta t)^2) & -1+4\xi\omega\Delta t & -\xi\omega\Delta t \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot \begin{Bmatrix} x_0 \\ x_{-1} \\ x_{-2} \end{Bmatrix} + \begin{Bmatrix} f_0\Delta t^2/m \\ 0 \\ 0 \end{Bmatrix} \tag{42}$$

算法的稳定性决定于传递矩阵,也即式(42)右边第1个矩阵的谱半径.当谱半径小于1的时候,算法是稳定的.通过数值分析,可得传递矩阵谱半径变化示意图,如图2,其稳定范围取决于阻尼比 ξ 和 $\Delta t/T$ 的值,其中 T 为结构的自振周期.表1中列出了不同方法下不同 ξ 值所对应的 $\Delta t/T$ 值.

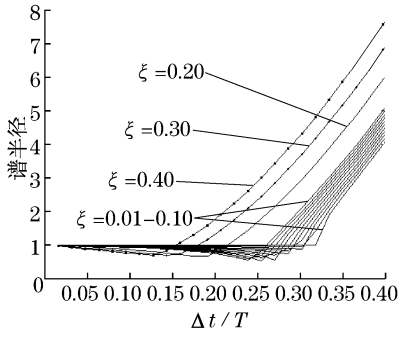


图2 本文算法传递矩阵谱半径变化示意

Fig.2 Spectral radius of the transferring matrix in three-step explicit displacement method

可见,在 $\xi\leqslant 0.10$ 时,本文算法的稳定性跟文献[1]和[4]相差不大,但是随着 ξ 增大,本文方法稳定性不如文献[1]和[4].

5 计算效率和所需存储空间

一种有竞争力的数值求解算法应该有较高的计算效率,需要较少的计算存储空间.本文从计算效率和所需计算存储空间2个方面将本文方法与其他方法进行比较.

5.1 计算效率

为了计算快速,式(35)可以改写为

$$\boldsymbol{x}_1 = \boldsymbol{M}^{-1}\boldsymbol{C}(-1.5\boldsymbol{x}_0 + 2\boldsymbol{x}_{-1} - 0.5\boldsymbol{x}_{-2})\Delta t + \boldsymbol{M}^{-1}\boldsymbol{K}(-\boldsymbol{x}_0) + 2(\boldsymbol{x}_0 - \boldsymbol{x}_{-1})\Delta t^2 + \boldsymbol{M}^{-1}\boldsymbol{F}_0\Delta t^2 \tag{43}$$

需要注意的是,1个对角矩阵与矩阵相乘,需要的只是 n^2 次乘法.而且计算 $\boldsymbol{M}^{-1}\boldsymbol{C}\boldsymbol{x}$,其中 $\boldsymbol{x}$ 为 n 维向量时,$(\boldsymbol{M}^{-1}\boldsymbol{C})\boldsymbol{x}$ 与 $\boldsymbol{M}^{-1}(\boldsymbol{C}\boldsymbol{x})$ 的计算效率是不一样的.如果先计算2个矩阵的乘积,再计算矩阵与向量

表1 不同算法稳定范围 $\Delta t/T$ 的比较

Tab.1 Stabilization field $\Delta t/T$ of different methods

计算方法	ξ 值										
	0.01	0.02	0.03	0.04	0.05	0.06	0.08	0.10	0.20	0.30	0.40
本文方法	0.32	0.32	0.30	0.30	0.29	0.29	0.27	0.25	0.22	0.19	0.16
文献[1]	0.32	0.32	0.31	0.31	0.31	0.30	0.30	0.30	0.29	0.28	0.28
文献[4]	0.32	0.31	0.31	0.30	0.29	0.29	0.28	0.27	0.26	0.21	0.18

的乘积,需要 $2n^2$ 次乘法和 n^2-2n+1 次加减法;而先计算矩阵与向量的乘积,再将计算得到的这个向量与矩阵相乘,需 n^2+n 次乘法和 n^2+1 次加减法.显然后面的计算效率更高.因此,笔者考虑各算法的计算效率的的时候,都是按计算效率最高的顺序来计算的.

分析式(43),计算$(-1.5\boldsymbol{x}_0+2\boldsymbol{x}_{-1}-0.5\boldsymbol{x}_{-2})$需要 $3n$ 次乘法运算和 $2n$ 次加减法运算.接着计算 $\boldsymbol{C}(-1.5\boldsymbol{x}_0+2\boldsymbol{x}_{-1}-0.5\boldsymbol{x}_{-2})$.这是 1 个 $n\times n$ 矩阵与 1 个 n 维向量的乘法,需 n^2 次乘法运算和$(n-1)^2$ 次加法运算.接着计算 $\boldsymbol{M}^{-1}\boldsymbol{C}(-1.5\boldsymbol{x}_0+2\boldsymbol{x}_{-1}-0.5\boldsymbol{x}_{-2})\Delta t$,这是 1 个对角矩阵与 1 个向量的运算,需要 n 次乘法运算,接着计算 $\boldsymbol{M}^{-1}\boldsymbol{C}(-1.5\boldsymbol{x}_0+2\boldsymbol{x}_{-1}-0.5\boldsymbol{x}_{-2})\Delta t$,这是 1 个向量乘以 1 个常数,需 n 次乘法运算,因此计算 $\boldsymbol{M}^{-1}\boldsymbol{C}(-1.5\boldsymbol{x}_0+2\boldsymbol{x}_{-1}-0.5\boldsymbol{x}_{-2})\Delta t$ 总共需要 n^2+5n 次乘法运算和 n^2+1 次加减法运算.同样可以得到计算 $\boldsymbol{M}^{-1}\boldsymbol{K}(-\boldsymbol{x}_0)+2(\boldsymbol{x}_0-\boldsymbol{x}_{-1})\Delta t^2$ 需要 n^2+4n 次乘法运算,n^2+1 次加减法运算,计算 $\boldsymbol{M}^{-1}\boldsymbol{F}_0\Delta t^2$ 需要 $2n$ 次乘法运算,最后得到的 3 个向量相加,也就是将向量 $\boldsymbol{M}^{-1}\boldsymbol{C}(-1.5\boldsymbol{x}_0+2\boldsymbol{x}_{-1}-0.5\boldsymbol{x}_{-2})\Delta t$,$\boldsymbol{M}^{-1}\boldsymbol{K}(-\boldsymbol{x}_0)+2(\boldsymbol{x}_0-\boldsymbol{x}_{-1})\Delta t^2$,$\boldsymbol{M}^{-1}\boldsymbol{F}_0\Delta t^2$ 相加,需要 $2n$ 次加减法运算,因此,计算一步总共需要 $2n^2+11n$ 次乘法运算,$2n^2+2n+2$ 次加减法运算.

文献[1]的必需计算步为

$$\begin{cases}\boldsymbol{x}_{i+1}=\dfrac{1}{2}\Delta t^2\boldsymbol{M}^{-1}\boldsymbol{F}_i+(\boldsymbol{I}-\dfrac{1}{2}\Delta t^2\boldsymbol{M}^{-1}\boldsymbol{K})\boldsymbol{x}_i+\\ \qquad(\Delta t\boldsymbol{I}-\dfrac{1}{2}\Delta t^2\boldsymbol{M}^{-1}\boldsymbol{C})\,\dot{\boldsymbol{x}}_i\\ \dot{\boldsymbol{x}}_{i+1}=\dfrac{1}{2}\Delta t\boldsymbol{M}^{-1}(\boldsymbol{F}_{i+1}+\boldsymbol{F}_i)+\dot{\boldsymbol{x}}_i-(\dfrac{1}{2}\Delta t\boldsymbol{M}^{-1}\boldsymbol{K}+\\ \qquad\boldsymbol{M}^{-1}\boldsymbol{C})\boldsymbol{x}_{i+1}-(\dfrac{1}{2}\Delta t\boldsymbol{M}^{-1}\boldsymbol{K}-\boldsymbol{M}^{-1}\boldsymbol{C})\boldsymbol{x}_i\end{cases}\tag{44}$$

共需 $4n^2+12n$ 次乘法,$4n^2+2n+4$ 次加减法.

文献[4]算法的必需步为

$$\begin{cases}\boldsymbol{x}_{i+1}=\Delta t^2\boldsymbol{M}^{-1}\boldsymbol{F}_i+(2\boldsymbol{I}-\Delta t^2\boldsymbol{M}^{-1}\boldsymbol{K})\boldsymbol{x}_i-\\ \qquad\boldsymbol{x}_{i-1}-\Delta t^2\boldsymbol{M}^{-1}\boldsymbol{C}\dot{\boldsymbol{x}}_i\\ \boldsymbol{x}_{i+1}=(3\boldsymbol{x}_{i+1}-4\boldsymbol{x}_i+\boldsymbol{x}_{i-1})/2\Delta t\end{cases}\tag{45}$$

共需 $2n^2+7n$ 次乘法和 $2n^2+2$ 次加减法.

文献[5]包含的必需步为

$$\begin{cases}\boldsymbol{x}_{i+1}=\boldsymbol{x}_i+\Delta t\,\dot{\boldsymbol{x}}_i+\dfrac{1}{2}\Delta t^2\,\ddot{\boldsymbol{x}}_i+\dfrac{1}{6}\Delta t^3\,\dddot{\boldsymbol{x}}_i\\ \boldsymbol{x}_{i+1}=\dot{\boldsymbol{x}}_i+\Delta t\,\ddot{\boldsymbol{x}}_i+\dfrac{1}{2}\Delta t^2\,\dddot{\boldsymbol{x}}_i\\ \boldsymbol{x}_{i+1}=\boldsymbol{M}^{-1}(\boldsymbol{F}_{i+1}-\boldsymbol{C}\dot{\boldsymbol{x}}_{i+1}-\boldsymbol{K}\boldsymbol{x}_{i+1})\\ \boldsymbol{x}_{i+1}=\boldsymbol{M}^{-1}(\dot{\boldsymbol{F}}_{i+1}-\boldsymbol{C}\ddot{\boldsymbol{x}}_{i+1}-\boldsymbol{K}\dot{\boldsymbol{x}}_{i+1})\end{cases}\tag{46}$$

共需 $4n^2+7n$ 次乘法运算,$4n^2+n+4$ 次加减法运算.

文献[6]包含的必需步为

$$\begin{cases}\boldsymbol{x}_{i+1}=\dfrac{\Delta t^2}{2}\boldsymbol{M}^{-1}\boldsymbol{F}_i+\Big(\boldsymbol{I}-\dfrac{\Delta t^2}{2}\boldsymbol{M}^{-1}\boldsymbol{K}\Big)\boldsymbol{x}_i+\\ \qquad\Big(\Delta t\boldsymbol{I}-\dfrac{\Delta t^2}{2}\boldsymbol{M}^{-1}\boldsymbol{C}\Big)\dot{\boldsymbol{x}}_i\\ \boldsymbol{x}_{i+1}=\beta_1\Delta t\boldsymbol{M}^{-1}(\boldsymbol{F}_{i+1}-\boldsymbol{F}_i)+(\boldsymbol{I}-\beta_1\Delta t^2\boldsymbol{M}^{-1}\boldsymbol{K})\cdot\\ \qquad\dot{\boldsymbol{x}}_i+\Delta t\Big(\boldsymbol{I}-\beta_1\Delta t\boldsymbol{M}^{-1}\boldsymbol{C}-\dfrac{\beta_1\Delta t^2}{2}\boldsymbol{M}^{-1}\boldsymbol{K}\Big)\ddot{\boldsymbol{x}}_i\\ \boldsymbol{x}_{i+1}=\boldsymbol{M}^{-1}(\boldsymbol{F}_{i+1}-\boldsymbol{F}_i)-\Delta t\boldsymbol{M}^{-1}\boldsymbol{K}\boldsymbol{x}_i+\\ \qquad\Big(\boldsymbol{I}-\Delta t\boldsymbol{M}^{-1}\boldsymbol{C}-\dfrac{\Delta t^2}{2}\boldsymbol{M}^{-1}\boldsymbol{K}\Big)\ddot{\boldsymbol{x}}_i\end{cases}\tag{47}$$

需要 $6n^2+19n$ 次乘法运算,$6n^2+2n+6$ 次加减法运算.可以看出,当 n 较大时,本文方法计算效率与文献[4]方法相当,进行同一步的计算所需时间分别是文献[1],[5]和[6]的 50%,50%和 33%.

5.2　所需存储空间

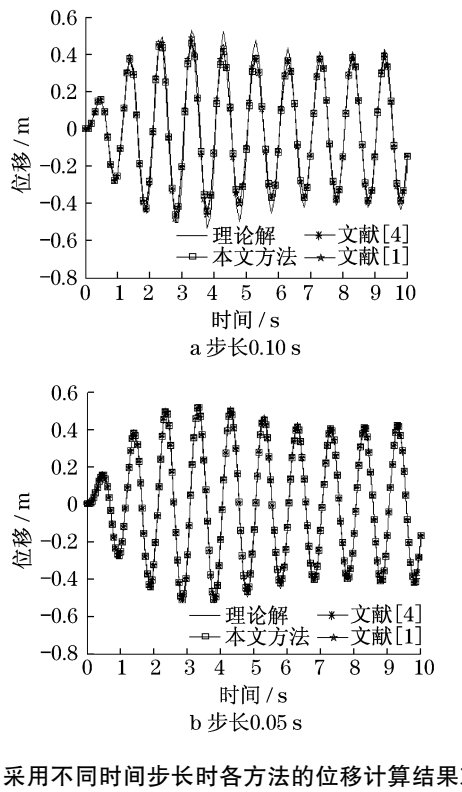
本文方法在计算过程中,计算结果存储只需要存储位移.文献[1],[4]能够计算下去,必须存储位移和速度.文献[6]能够计算下去,必须存储位移、速度和加速度,而文献[5]则需要存储位移、速度、加速度和位移的三阶导数.因此本文需要存储的数据只是方法[1]和[4]的 50%,方法[6]的 33%,方法[5]的 25%,显然按本文方法进行下去所需的存储空间最少.

6　算例

计算一个单自由度有阻尼体系,在荷载 $f(t)=10\sin(4\pi t)$作用下的反应,其中计算参数为:$k=100.0\ \mathrm{N\cdot m}$,$m=2.0\ \mathrm{kg}$,阻尼比 $\xi=0.05\ \mathrm{N\cdot s\cdot m^{-1}}$,初始位移 $x_0=0\ \mathrm{m}$,初始速度为 $\dot{x}_0=0\ \mathrm{m\cdot s^{-1}}$.

分别用本文方法和文献[1]、文献[4]方法进行了动力时程计算,如图 3.可见,当时间步长为 0.10 s

时,3种方法计算得到的位移曲线与理论解基本重合;当时间步长为0.05 s时,本文方法的计算结果与精确解几乎完全吻合.



a 步长0.10 s

b 步长0.05 s

图3 采用不同时间步长时各方法的位移计算结果对比

Fig.3 Displacement results of different methods

7 结语

本文提出的三步二阶精度显式位移法,可以认为是中心差分法的一种变化格式,也即加速度仍采用中心差分格式近似,但速度采用了单边三点差分格式近似,所以本方法也可以称为中心-偏心差分法.对导数的高精度差分近似进行了阐述,得出了差分格式中增加1个计算点可以提高1阶精度的结论,这可以用于其他需要高精度数值微分的研究中.提出了一种只含位移的显式动力时程计算方法的思路.中心-偏心差分法与同类的显式算法相比具有较高的计算效率,而且需要的计算存储空间最小.

参考文献:

[1] 李小军,廖振鹏,杜修力.有阻尼体系动力问题的一种显式差分解法[J].地震工程与工程振动,1992,12(4):74.
LI Xiaojun,LIAO Zhenpeng,DU Xiuli. An explicit finite difference method for viscoelastic dynamic problem [J]. Earthquake Engineering and Engineering Vibration,1992,12(4):74.

[2] 李小军,廖振鹏.非线性结构动力方程求解的显式差分格式的特性分析[J].工程力学,1993(3):141.
LI Xiaojun ,LIAO Zhenpeng. The analysis of the properties of an explicit difference method for solving the nonlinear structural dynamic equations[J]. Engineering Mechanics,1993(3):141.

[3] 杜修力,王进廷.阻尼弹性结构动力计算的显式差分法[J].工程力学,2000(5):37.
DU Xiuli,WANG jinting. An explicit difference formulation of dynamic response calculation of elastic structure with damping [J]. Engineering Mechanics,2000(5):37.

[4] 王进廷,杜修力.有阻尼体系动力分析的一种显式差分法[J].工程力学,2002(3):109.
WANG Jinting,DU Xiuli. An explicit difference method for dynamic analysis of a structure system with damping [J]. Engineering Mechanics,2002(3):109.

[5] 张晓志,程岩,谢礼立.结构动力反应分析的三阶显式方法[J].地震工程与工程振动,2002(3):1.
ZHANG Xiaozhi,CHENG Yan , XIE Lili. A new explicit solution of dynamic response analysis[J]. Earthquake Engineering and Engineering Vibration,2002(3):1.

[6] 陈学良,金星,陶夏新.求解加速度反应的显式积分格式研究[J].地震工程与工程振动,2006(5):60.
CHEN Xueliang,JIN Xing ,TAO Xiaxin. Study on explicit integration formula for dynamic acceleration response [J]. Earthquake Engineering and Engineering Vibration,2006(5):60.