Numerical Simulation of Dynamic Systems VII

Prof. Dr. François E. Cellier
Department of Computer Science
ETH Zurich

March 19, 2013

In Search of Stiffly-stable Methods

An interpolation polynomial of order n can be written as:

$$p(s) = a_0 + a_1\ s + a_2\ s^2 + a_3\ s^3 + \cdots + a_n\ s^n$$

where s is the *normalized time* variable introduced earlier.

Its derivative with respect to time t can be formulated as:

$$h \cdot \dot{p}(s) = a_1 + 2a_2\ s + 3a_3\ s^2 + \cdots + n\ a_n\ s^{n-1}$$

In the case of the BDF3 algorithm ($n = 3$), we know that:

$$\begin{aligned} h \cdot \dot{p}(s = +1) &= h \cdot f_{k+1} \\ p(s = 0) &= x_k \\ p(s = -1) &= x_{k-1} \\ p(s = -2) &= x_{k-2} \end{aligned}$$

Therefore:

$$\begin{aligned} h \cdot f_{k+1} &= a_1 + 2a_2 + 3a_3 \\ x_k &= a_0 \\ x_{k-1} &= a_0 - a_1 + a_2 - a_3 \\ x_{k-2} &= a_0 - 2a_1 + 4a_2 - 8a_3 \end{aligned}$$

In Search of Stiffly-stable Methods II

In matrix/vector form:

$$\begin{pmatrix} h \cdot f_{k+1} \\ x_k \\ x_{k-1} \\ x_{k-2} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 2 & 3 \\ 1 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 \\ 1 & -2 & 4 & -8 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

We can determine the values of the parameters and calculate:

$$x_{k+1} = p(s = +1) = a_0 + a_1 + a_2 + a_3$$

Using a computer algebra tool, such as **Maple**, we obtain:

$$x_{k+1} = \frac{6}{11} h \cdot f_{k+1} + \frac{18}{11} x_k - \frac{9}{11} x_{k-1} + \frac{2}{11} x_{k-2}$$

i.e., the coefficients of the BDF3 algorithm found earlier using another approach.

We can make use of this alternate technique to search for *other linear multi-step methods for the numerical simulation of dynamic systems*.

In Search of Stiffly-stable Methods III

We might suspect that the extrapolation didn't work so well until now because of the *long tail* of the interpolation polynomial. It could be a good idea to reduce the length of the tail.

Consequently, we shall design a *sixth-order linear multi-step algorithm* that passes through three state values and four state derivative values. In this way, we can reduce the length of the tail by two steps:

$$\begin{pmatrix} h \cdot f_{k+1} \\ x_k \\ h \cdot f_k \\ x_{k-1} \\ h \cdot f_{k-1} \\ x_{k-2} \\ h \cdot f_{k-2} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 0 & 1 & -2 & 3 & -4 & 5 & -6 \\ 1 & -2 & 4 & -8 & 16 & -32 & 64 \\ 0 & 1 & -4 & 12 & -32 & 80 & -192 \end{pmatrix} \cdot \mathbf{a}$$

We obtain a beautiful and totally unknown *sixth-order accurate algorithm*:

$$\mathbf{x_{k+1}} = \frac{3}{11} h \cdot \mathbf{f_{k+1}} - \frac{27}{11}\mathbf{x_k} + \frac{27}{11} h \cdot \mathbf{f_k} + \frac{27}{11}\mathbf{x_{k-1}} + \frac{27}{11} h \cdot \mathbf{f_{k-1}} + \mathbf{x_{k-2}} + \frac{3}{11} h \cdot \mathbf{f_{k-2}}$$

Unfortunately, this algorithm is unstable everywhere.

In Search of Stiffly-stable Methods IV

Evidently, shortening the tail was a bad idea.

Now, we shall again permit the use of state values and state derivative values up to $t = t_{k-5}$, i.e., until $s = -5$. We are only interested in sixth-order accurate implicit methods. Thus, all of the considered methods should include the state derivative value $\mathbf{f_{k+1}}$.

For a sixth-order accurate method, we need to choose 6 terms out of the available 12. Hence there exist 924 candidate methods.

Most of the 924 candidate methods are entirely unstable. Others behave like Adams-Moulton. Only six out of the 924 methods have an intersection of their respective stability domains with the positive real axis.

In Search of Stiffly-stable Methods V

The six remaining methods are listed below:

$$
\begin{aligned}
(a)\quad \mathbf{x}_{k+1} &= \frac{20}{49} h \cdot \mathbf{f}_{k+1} + \frac{120}{49}\mathbf{x}_k - \frac{150}{49}\mathbf{x}_{k-1} + \frac{400}{147}\mathbf{x}_{k-2} - \frac{75}{49}\mathbf{x}_{k-3} + \frac{24}{49}\mathbf{x}_{k-4} - \frac{10}{147}\mathbf{x}_{k-5} \\
(b)\quad \mathbf{x}_{k+1} &= \frac{308}{745} h \cdot \mathbf{f}_{k+1} + \frac{1776}{745}\mathbf{x}_k - \frac{414}{149}\mathbf{x}_{k-1} + \frac{944}{447}\mathbf{x}_{k-2} - \frac{87}{149}\mathbf{x}_{k-3} - \frac{288}{745} h \cdot \mathbf{f}_{k-4} - \frac{2}{15}\mathbf{x}_{k-5} \\
(c)\quad \mathbf{x}_{k+1} &= \frac{8820}{21509} h \cdot \mathbf{f}_{k+1} + \frac{52200}{21509}\mathbf{x}_k - \frac{63900}{21509}\mathbf{x}_{k-1} + \frac{400}{157}\mathbf{x}_{k-2} - \frac{28575}{21509}\mathbf{x}_{k-3} + \frac{6984}{21509}\mathbf{x}_{k-4} \\
&\quad + \frac{600}{21509} h \cdot \mathbf{f}_{k-5} \\
(d)\quad \mathbf{x}_{k+1} &= \frac{179028}{432845} h \cdot \mathbf{f}_{k+1} + \frac{206352}{86569}\mathbf{x}_k - \frac{34452}{12367}\mathbf{x}_{k-1} + \frac{26704}{12367}\mathbf{x}_{k-2} - \frac{65547}{86569}\mathbf{x}_{k-3} - \frac{83808}{432845} h \cdot \mathbf{f}_{k-4} \\
&\quad + \frac{24}{581} h \cdot \mathbf{f}_{k-5} \\
(e)\quad \mathbf{x}_{k+1} &= \frac{12}{29} h \cdot \mathbf{f}_{k+1} + \frac{1728}{725}\mathbf{x}_k - \frac{81}{29}\mathbf{x}_{k-1} + \frac{64}{29}\mathbf{x}_{k-2} - \frac{27}{29}\mathbf{x}_{k-3} + \frac{97}{725}\mathbf{x}_{k-5} + \frac{12}{145} h \cdot \mathbf{f}_{k-5} \\
(f)\quad \mathbf{x}_{k+1} &= \frac{30}{71} h \cdot \mathbf{f}_{k+1} + \frac{162}{71}\mathbf{x}_k - \frac{675}{284}\mathbf{x}_{k-1} + \frac{100}{71}\mathbf{x}_{k-2} - \frac{54}{71}\mathbf{x}_{k-4} + \frac{127}{284}\mathbf{x}_{k-5} + \frac{15}{71} h \cdot \mathbf{f}_{k-5}
\end{aligned}
$$

We are already familiar with method (a), as this is the already familiar *BDF6 algorithm*.

The Numerical Stability Domains

Two of the six methods are useless, because they exhibit problems with their numerical stability domains. I drew the numerical stability domains of the remaining four algorithms.

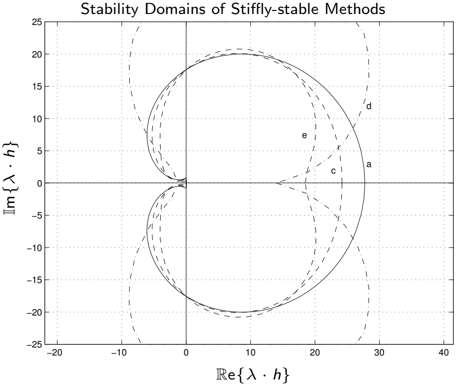

Figure: Stability domains of some stiffly-stable algorithms

Comparison of the Algorithms

How can we decide, which of the four algorithms is the best?

On the one hand, we may analyze the "damage" that we incurred because of the unstable region to the left of the imaginary axis.

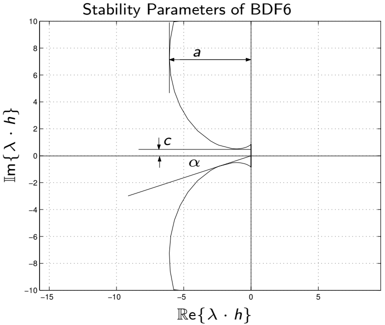

Figure: Stability parameters of a stiffly-stable algorithm

Comparison of the Algorithms II

- It is possible to define an angle α with the negative real axis that specifies a *region guaranteed to be stable*.
- Algorithms that aren't completely A-stable can still be characterized as *A(α)-stable*. The BDF6 algorithm is A(α)-stable with an angle of $\alpha = 19^{o}$.
- It is also possible to measure the distances *a* from the negative real axis and *c* from the imaginary axis that exclude all unstable regions.
- Together, the three parameters characterize well the "damage" that we incurred because of the unstable region to the left of the complex $\boxed{\lambda \cdot h}$ plane.

Comparison of the Algorithms III

On the other hand, we may also wish to compare the *numerical accuracy of the algorithms*.

All linear multi-step algorithms can be written in the form:

$$\mathbf{x_{k+1}} = \sum_{i=0}^{\ell} a_i \cdot \mathbf{x_{k-i}} + \sum_{i=-1}^{\ell} b_i \cdot h \cdot \mathbf{f_{k-i}}$$

Shifting the equation by ℓ steps into the future:

$$\sum_{i=0}^{m} \alpha_i \cdot \mathbf{x_{k+i}} + h \cdot \sum_{i=0}^{m} \beta_i \cdot \mathbf{f_{k+i}} = 0$$

We can develop $\mathbf{x_{k+i}}$ and $\mathbf{f_{k+i}}$ into Taylor series around $\mathbf{x_k}$ and $\mathbf{f_k}$, and come up with an expression in $\mathbf{x_k}$ and its derivatives:

$$c_0 \cdot \mathbf{x_k} + c_1 \cdot h \cdot \dot{\mathbf{x}}_{\mathbf{k}} + \cdots + c_q \cdot h^q \cdot \mathbf{x_k^{(q)}} + \ldots$$

where $\mathbf{x_k^{(q)}}$ is the q^{th} time derivative of $\mathbf{x_k}$.

Comparison of the Algorithms IV

We find:

$$c_0 = \sum_{i=0}^{m} \alpha_i$$

$$c_1 = \sum_{i=0}^{m} (i \cdot \alpha_i - \beta_i)$$

$$\vdots$$

$$c_q = \sum_{i=0}^{m} \left(\frac{1}{q!} \cdot i^q \cdot \alpha_i - \frac{1}{(q-1)!} \cdot i^{q-1} \cdot \beta_i \right), \quad q = 2, 3, \ldots$$

Since the function that has been developed into a Taylor series is the zero function, all of these coefficients ought to be equal to zero. However, since the approximation is only n^{th}-order accurate, the coefficients for $q > n$ may be different from zero. Hence we can define the dominant of those coefficients as the *error coefficient* of the multi-step integration algorithm.

Comparison of the Algorithms V

Hence we can define an *error coefficient* of the method as:

$$c_{err} = \sum_{i=0}^{m} \left(\frac{1}{(n+1)!} \cdot i^{n+1} \cdot \alpha_i - \frac{1}{n!} \cdot i^n \cdot \beta_i \right)$$

Comparing the error coefficients of the BDF methods with those of the Adams methods, we notice that the error coefficients of the Adams methods are quite a bit smaller than those of the BDF methods.

Consequently, if we use a linear multi-step algorithm for the *simulation of a non-stiff system*, we obtain higher precision for the same step size, h, (i.e., for the same "price"), with **Adams-Moulton** than with **BDF**.

For this reason, it is useful to include the error coefficient in the evaluation of the overall quality of a linear multi-step algorithm.

Comparison of the Algorithms VI

It is also useful to look at the *damping plot* of each algorithm.

In the analysis of the damping properties of single-step algorithms, it sufficed to look at the scalar system. The damping factor was defined as:

$$\hat{\sigma}_d = -\log(\mathrm{abs}(f))$$

In the analysis of the damping properties of multi-step algorithms, this approach won't work any longer, because already the scalar system is characterized by an **F**-matrix of size 3×3.

Hence we need to extend the definition:

$$\hat{\sigma}_d = -\log(\max(\mathrm{abs}(\mathrm{eig}(\mathbf{F}))))$$

We shall now draw not only the *linear damping plot*, but also the *logarithmic damping plot*.

The Damping Plots of the Algorithms

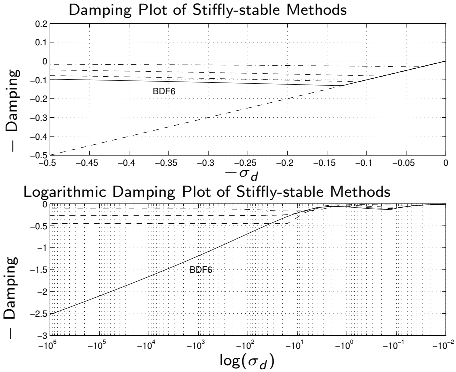

Figure: Damping plot of BDF6 and other 6^{th}-order stiffly-stable methods

We notice that only the BDF6 algorithm is L(α)-stable. None of the other algorithms exhibits this property.

Comparison of the Algorithms VII

- BDF6 apparently is the best of all sixth-order linear multi-step algorithms for the simulation of stiff systems.

- All algorithms that survived the search exploit the full available range of support values up to t_{k-5}. Apparently shortening the length of the tail of the algorithm was a very bad idea.

- None of those algorithms that survived the search use more than a single value of a state derivative from the past. It seems to be a bad idea trying to use state derivatives of the past in the design of algorithms for the simulation of stiff systems.

Extending the Search

- It might possibly be a good idea to lengthen the tail of the algorithm. As we already decided that state derivatives from the past are not useful, we shall limit our new search in such a way that we shall only consider algorithms that are making use of state values from the past only. We shall extend the length of the tail to t_{k-11}.
- Once again, there exist 924 potential algorithms of order six. Among them, 314 exhibit characteristics similar to those of the BDF6 algorithm.

Their five performance parameters are:

BDF6	Other stiffly-stable methods
$\alpha = 19^\circ$	$\alpha \in [19^\circ, 48^\circ]$
$a = -6.0736$	$a \in [-6.0736, -0.6619]$
$c = 0.5107$	$c \in [0.2250, 0.8316]$
$c_{err} = -0.0583$	$c_{err} \in [-7.4636, -0.0583]$
$as.reg. = -0.14$	$as.reg. \in [-0.30, -0.01]$

In order to evaluate the performance of these methods quantitatively, we require a performance index:

$$P.I._i = \frac{|\alpha_i|}{\|\alpha\|} - \frac{|a_i|}{\|a\|} + \frac{|c_i|}{\|c\|} - k \cdot \frac{|c_{err_i}|}{\|c_{err}\|} + \frac{|as.reg._i|}{\|as.reg.\|} = max!$$

We choose $k = 20$ to indicate that a small error coefficient is very important.

Extending the Search II

The best three algorithms are:

$$\mathbf{x_{k+1}} = \frac{72}{167} h \cdot \mathbf{f_{k+1}} + \frac{2592}{1169}\mathbf{x_k} - \frac{2592}{1169}\mathbf{x_{k-1}} + \frac{1152}{835}\mathbf{x_{k-2}} - \frac{324}{835}\mathbf{x_{k-3}} + \frac{81}{5845}\mathbf{x_{k-7}} - \frac{32}{5845}\mathbf{x_{k-8}}$$

$$\mathbf{x_{k+1}} = \frac{420}{977} h \cdot \mathbf{f_{k+1}} + \frac{19600}{8793}\mathbf{x_k} - \frac{2205}{977}\mathbf{x_{k-1}} + \frac{1400}{977}\mathbf{x_{k-2}} - \frac{1225}{2931}\mathbf{x_{k-3}} + \frac{40}{2931}\mathbf{x_{k-6}} - \frac{7}{8793}\mathbf{x_{k-9}}$$

$$\mathbf{x_{k+1}} = \frac{44}{103} h \cdot \mathbf{f_{k+1}} + \frac{5808}{2575}\mathbf{x_k} - \frac{242}{103}\mathbf{x_{k-1}} + \frac{484}{309}\mathbf{x_{k-2}} - \frac{363}{721}\mathbf{x_{k-3}} + \frac{242}{7725}\mathbf{x_{k-5}} - \frac{4}{18025}\mathbf{x_{k-10}}$$

with the following performance parameters:

BDF6	SS6a	SS6b	SS6c
$\alpha = 19^\circ$ $a = -6.0736$ $c = 0.5107$ $c_{err} = -0.0583$ $as.reg. = -0.14$	$\alpha = 45^\circ$ $a = -2.6095$ $c = 0.7994$ $c_{err} = -0.1478$ $as.reg. = -0.21$	$\alpha = 44^\circ$ $a = -2.7700$ $c = 0.8048$ $c_{err} = -0.1433$ $as.reg. = -0.21$	$\alpha = 43^\circ$ $a = -3.0839$ $c = 0.8156$ $c_{err} = -0.1343$ $as.reg. = -0.21$

Extending the Search III

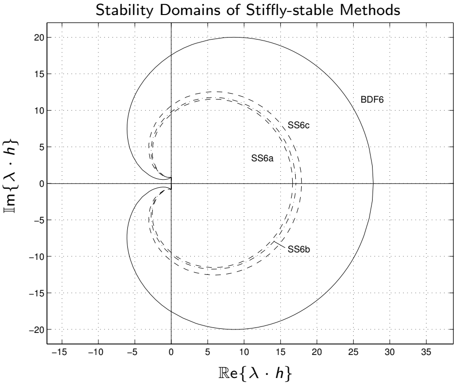

Figure: Stability domains of some stiffly-stable algorithms

Extending the Search IV

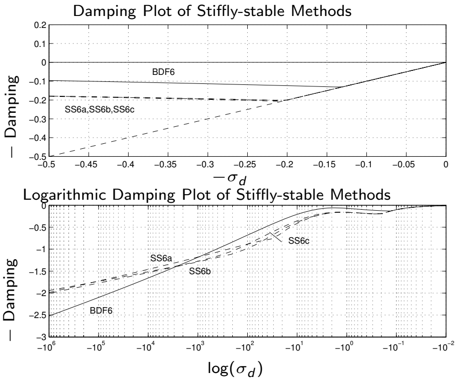

Figure: Damping plot of BDF6 and other 6^{th}-order stiffly-stable methods

Extending the Search V

- **Finally, we encountered a few stiffly-stable sixth-order multi-step methods that are better than BDF6.**
- The new algorithms can also be used for the *simulation of systems with oscillatory behavior*, as their angle α is quite a bit larger than that of BDF6.
- The most important characteristic of these three algorithms is their *asymptotic region*, which is almost 50% larger than that of BDF6. This enables us to use larger step sizes during the simulation.
- The *error coefficient*, c_{err}, of these new methods is a bit larger than that of BDF6, but this is not important. The integration step size of BDF6 is much more frequently limited by the asymptotic region than by the magnitude of c_{err}.

High-order Backward Difference Formulae

Using the same methodology, we may be able to find *higher-order backward difference formulae*. To this end, we enlarged the length of the tail to t_{k-13} and searched for *7^{th}-order accurate BDF methods*. Thus, we have to choose seven support values from an available number of 14.

Of the possible 3432 algorithms, 762 possess properties similar to BDF6, i.e., they are *$A(\alpha)$-stable* and also *L-stable*. The search was limited to algorithms with $\alpha \geq 10°$.

Their five performance parameters are:

BDF6	7^{th}-order stiffly-stable methods
$\alpha = 19°$	$\alpha \in [10°, 48°]$
$a = -6.0736$	$a \in [-6.1261, -0.9729]$
$c = 0.5107$	$c \in [0.0811, 0.7429]$
$c_{err} = -0.0583$	$c_{err} \in [-6.6498, -0.1409]$
$as.reg. = -0.14$	$as.reg. \in [-0.23, -0.01]$

The smallest error coefficient is now almost three times larger than in the case of the 6^{th}-order algorithms. The other parameters are comparable in their ranges with those of the 6^{th}-order algorithms.

High-order Backward Difference Formulae II

The best three algorithms are:

$$
\begin{aligned}
\mathbf{x_{k+1}} = & \frac{5148}{12161} h \cdot \mathbf{f_{k+1}} + \frac{552123}{243220} \mathbf{x_k} - \frac{200772}{85127} \mathbf{x_{k-1}} + \frac{184041}{121610} \mathbf{x_{k-2}} \\
& - \frac{184041}{425635} \mathbf{x_{k-3}} + \frac{20449}{1702540} \mathbf{x_{k-8}} - \frac{4563}{851270} \mathbf{x_{k-10}} + \frac{99}{121610} \mathbf{x_{k-12}} \\
\mathbf{x_{k+1}} = & \frac{234}{551} h \cdot \mathbf{f_{k+1}} + \frac{13689}{6061} \mathbf{x_k} - \frac{492804}{212135} \mathbf{x_{k-1}} + \frac{4056}{2755} \mathbf{x_{k-2}} \\
& - \frac{4563}{11020} \mathbf{x_{k-3}} + \frac{169}{19285} \mathbf{x_{k-8}} - \frac{507}{121220} \mathbf{x_{k-11}} + \frac{54}{30305} \mathbf{x_{k-12}} \\
\mathbf{x_{k+1}} = & \frac{3276}{7675} h \cdot \mathbf{f_{k+1}} + \frac{17199}{7675} \mathbf{x_k} - \frac{191646}{84425} \mathbf{x_{k-1}} + \frac{596232}{422125} \mathbf{x_{k-2}} \\
& - \frac{74529}{191875} \mathbf{x_{k-3}} + \frac{1183}{191875} \mathbf{x_{k-8}} - \frac{882}{422125} \mathbf{x_{k-12}} + \frac{2106}{2110625} \mathbf{x_{k-13}}
\end{aligned}
$$

with the following performance parameters:

BDF6	SS7a	SS7b	SS7c
$\alpha = 19^\circ$	$\alpha = 37^\circ$	$\alpha = 39^\circ$	$\alpha = 35^\circ$
$a = -6.0736$	$a = -3.0594$	$a = -2.9517$	$a = -3.2146$
$c = 0.5107$	$c = 0.6352$	$c = 0.6664$	$c = 0.6331$
$c_{err} = -0.0583$	$c_{err} = -0.3243$	$c_{err} = -0.3549$	$c_{err} = -0.3136$
$as.reg. = -0.14$	$as.reg. = -0.15$	$as.reg. = -0.16$	$as.reg. = -0.15$

High-order Backward Difference Formulae III

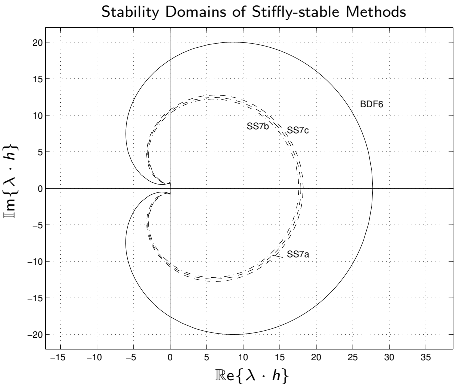

Figure: Stability domains of BDF6 and a set of 7^{th}-order stiffly-stable algorithms

High-order Backward Difference Formulae IV

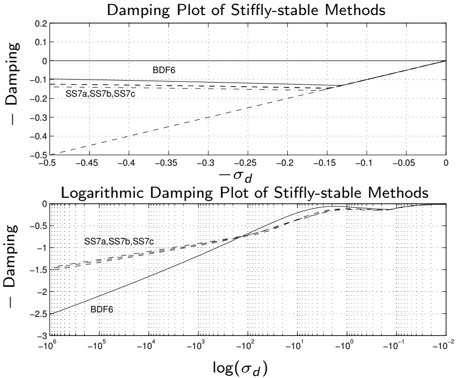

Figure: Damping plots of BDF6 and a set of 7^{th}-order stiffly-stable algorithms

High-order Backward Difference Formulae V

We can now proceed to 8^{th}*-order accurate algorithms*. We searched for algorithms with tails reaching all the way back to t_{k-15}. Hence we had to choose eight support values out of a list of 16 candidates. Of the possible 12870 algorithms, 493 exhibit properties similar to those of BDF6.

Their five performance parameters are:

BDF6	8^{th}–order stiffly–stable methods
$\alpha = 19^o$	$\alpha \in [10^o, 48^o]$
$a = -6.0736$	$a \in [-5.3881, -1.4382]$
$c = 0.5107$	$c \in [0.0859, 0.6485]$
$c_{err} = -0.0583$	$c_{err} \in [-6.4014, -0.4416]$
$as.reg. = -0.14$	$as.reg. \in [-0.16, -0.01]$

The smallest error coefficient has unfortunately again grown by about a factor of three, and this time around, also the largest asymptotic region has begun to shrink.

High-order Backward Difference Formulae VI

The two best algorithms are:

$$\mathbf{x_{k+1}} = \frac{112}{267} h \cdot \mathbf{f_{k+1}} + \frac{71680}{31239}\mathbf{x_k} - \frac{2800}{1157}\mathbf{x_{k-1}} + \frac{179200}{114543}\mathbf{x_{k-2}} - \frac{3920}{8811}\mathbf{x_{k-3}} + \frac{112}{12015}\mathbf{x_{k-9}} - \frac{160}{12727}\mathbf{x_{k-13}} + \frac{7168}{572715}\mathbf{x_{k-14}} - \frac{35}{10413}\mathbf{x_{k-15}}$$

$$\mathbf{x_{k+1}} = \frac{208}{497} h \cdot \mathbf{f_{k+1}} + \frac{216320}{93933}\mathbf{x_k} - \frac{93600}{38269}\mathbf{x_{k-1}} + \frac{16640}{10437}\mathbf{x_{k-2}} - \frac{67600}{147609}\mathbf{x_{k-3}} + \frac{5408}{469665}\mathbf{x_{k-9}} - \frac{1280}{147609}\mathbf{x_{k-12}} + \frac{3328}{574035}\mathbf{x_{k-14}} - \frac{65}{31311}\mathbf{x_{k-15}}$$

with the following performance parameters:

BDF6	SS8a	SS8b
$\alpha = 19^o$ $a = -6.0736$ $c = 0.5107$ $c_{err} = -0.0583$ $as.reg. = -0.14$	$\alpha = 35^o$ $a = -3.2816$ $c = 0.5779$ $c_{err} = -0.9322$ $as.reg. = -0.14$	$\alpha = 35^o$ $a = -3.4068$ $c = 0.5456$ $c_{err} = -0.8636$ $as.reg. = -0.13$

Their stability domains and damping plots look almost identical to those of the 7^{th}-order algorithms.

High-order Backward Difference Formulae VII

Let us now proceed to 9^{th}-*order accurate algorithms*. We decided to search for algorithms with their tails reaching back as far as t_{k-17}. Hence we had to choose 9 support values from a list of 18 candidates. Of the 48620 candidate algorithms, only 152 exhibit properties similar to those of BDF6.

Their five performance parameters are:

BDF6	9^{th}–order stiffly–stable methods
$\alpha = 19^{o}$	$\alpha \in [10^{o}, 32^{o}]$
$a = -6.0736$	$a \in [-5.0540, -2.4730]$
$c = 0.5107$	$c \in [0.0625, 0.4991]$
$c_{err} = -0.0583$	$c_{err} \in [-5.9825, -1.2492]$
$as.reg. = -0.14$	$as.reg. \in [-0.10, -0.02]$

The smallest error coefficient has once again grown by about a factor of three, and also the largest asymptotic region has now shrunk significantly.

High-order Backward Difference Formulae VIII

The two best algorithms are:

$$\mathbf{x_{k+1}} = \frac{4080}{9947} h \cdot \mathbf{f_{k+1}} + \frac{165240}{69629} \mathbf{x_k} - \frac{16854480}{6336239} \mathbf{x_{k-1}} + \frac{1664640}{905177} \mathbf{x_{k-2}} - \frac{5618160}{9956947} \mathbf{x_{k-3}} + \frac{23120}{1462209} \mathbf{x_{k-8}} - \frac{332928}{9956947} \mathbf{x_{k-14}} + \frac{351135}{6336239} \mathbf{x_{k-15}} - \frac{29160}{905177} \mathbf{x_{k-16}} + \frac{1360}{208887} \mathbf{x_{k-17}}$$

$$\mathbf{x_{k+1}} = \frac{1904}{4651} h \cdot \mathbf{f_{k+1}} + \frac{719712}{302315} \mathbf{x_k} - \frac{62424}{23255} \mathbf{x_{k-1}} + \frac{6214656}{3325465} \mathbf{x_{k-2}} - \frac{873936}{1511575} \mathbf{x_{k-3}} + \frac{18496}{1046475} \mathbf{x_{k-8}} - \frac{249696}{16627325} \mathbf{x_{k-13}} + \frac{7803}{302315} \mathbf{x_{k-15}} - \frac{6048}{302315} \mathbf{x_{k-16}} + \frac{952}{209295} \mathbf{x_{k-17}}$$

with the following performance parameters:

BDF6	SS9a	SS9b
$\alpha = 19^o$ $a = -6.0736$ $c = 0.5107$ $c_{err} = -0.0583$ $as.reg. = -0.14$	$\alpha = 18^o$ $a = -4.3280$ $c = 0.3957$ $c_{err} = -1.7930$ $as.reg. = -0.10$	$\alpha = 18^o$ $a = -4.3321$ $c = 0.3447$ $c_{err} = -1.6702$ $as.reg. = -0.08$

High-order Backward Difference Formulae IX

- We were able to find *backward difference formulae of orders 7..9*.
- Unfortunately, these methods exhibit once again *smaller α angles*. For orders seven and eight, we could still find methods with $\alpha = 35^{o}$. For order nine, the largest angle found was $\alpha = 18^{o}$.
- Also the *asymptotic region* is shrinking. For orders seven and eight, we could still find methods with $reg.as. = -0.14$. For order nine, the largest asymptotic region found was $reg.as. = -0.10$.
- The *error coefficient*, c_{err}, also grows rapidly. Yet, it makes little sense to compare error coefficients of methods of different approximation orders with each other. It only makes sense to compare the error coefficients of methods of identical orders.
- Methods of such high orders of approximation accuracy are useful for the simulation of *celestial mechanics* problems. However in methods of such high orders, the *roundoff error* in double precision is already larger than the *truncation error* on a 32-bit architecture. These algorithms should therefore be used on a 64-bit machine only or alternatively, they should be implemented in *quadruple precision*.

Conclusions

In this presentation, we demonstrated how new multi-step algorithms can be found that are of interest for the numerical simulation of stiff systems.

We developed a set of performance parameters and a performance index that help us to quickly compare many thousands of different candidate methods and find those among them that look most promising.

For the first time, we were able to discover stable BDF algorithms of orders higher than six. In particular, we found several promising BDF algorithms of orders 7..9.

References

1. Hermann, Klaus (1995), *Solution of Stiff Systems Described by Ordinary Differential Equations by Means of Regression Backward Difference Formulae (RBDF)*, MS Thesis, Dept. of Electrical & Computer Engineering, University of Arizona, Tucson, AZ.