Zur Lösung differential-algebraischer Gleichungssysteme

Eine Einführung von M. Burkhardt und R. Seifried

1 Differential-algebraische Gleichungen

Die implizite Differentialgleichung

$$\boldsymbol{h}(t, \boldsymbol{x}, \dot{\boldsymbol{x}}) = \boldsymbol{0} \tag{1}$$

beschreibt die Änderung $\dot{\boldsymbol{x}}$ einer n-dimensionalen Größe $\boldsymbol{x}$ über der Zeit t. Ist die Jacobi-Matrix $\dfrac{\partial \boldsymbol{h}}{\partial \dot{\boldsymbol{x}}}$ regulär, so handelt es sich bei Gleichung (1) um ein System gewöhnlicher Differentialgleichungen. Ist diese Jacobi-Matrix jedoch singulär, so spricht man von einem differential-algebraischen System. Häufig ist es möglich, diese implizite Form (1) durch die semi-explizite Form

$$\boldsymbol{I} \cdot \dot{\boldsymbol{x}} = \boldsymbol{k}(t, \boldsymbol{x}) \tag{2}$$

darzustellen. Die darin enthaltene Matrix $\boldsymbol{I}$ ist im Falle von differential-algebraischen Gleichungen singulär und im Falle einer gewöhnlichen Differentialgleichung regulär. Diese Form eignet sich jedoch nicht zur numerischen Lösung. Durch eine Koordinatentransformation lässt sich Gleichung (2) durch

$$\dot{\boldsymbol{y}} = \boldsymbol{f}(t, \boldsymbol{y}, \boldsymbol{u}) \tag{3a}$$
$$\boldsymbol{0} = \boldsymbol{c}(t, \boldsymbol{y}, \boldsymbol{u}) \tag{3b}$$

ausdrücken. Die Trennung in eine Differentialgleichung (3a) und eine algebraische Gleichung (3b) führt auf die explizite Darstellungsform, die zur numerischen Lösung herangezogen wird.

2 Runge-Kutta-Verfahren

Bei Runge-Kutta-Verfahren handelt es sich um Einschrittverfahren zur numerische Lösung von Differentialgleichungen der Form

$$\dot{\boldsymbol{y}} = \boldsymbol{f}(t, \boldsymbol{y}), \tag{4}$$

die sich durch die Berechnungsvorschrift

$$\boldsymbol{g}_i = \boldsymbol{y}_k + h \sum_{j=1}^{s} a_{ij} \boldsymbol{f}(t_k + c_j h, \boldsymbol{g}_j) \qquad i = 1 \dots s \tag{5a}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + h \sum_{j=1}^{s} b_j \boldsymbol{f}(t_k + c_j h, \boldsymbol{g}_j) \tag{5b}$$

ausdrücken lassen. Dies ist eine Approximation der Lösung der Differentialgleichung vom Zeitpunkt t_k bis zum Zeitpunkt $t_{k+1} = t_k + h$, wobei h die Schrittweite kennzeichnet. Dieses Intervall wird nun an den Stützstellen $\boldsymbol{g}_i$ mit der Differentialgleichung in Einklang gebracht. Mit Hilfe des Butcher-Blocks

$$\begin{array}{c|c} \boldsymbol{c} & \boldsymbol{A} \\ \hline & \boldsymbol{b} \end{array} = \begin{array}{c|ccc} c_1 & a_{11} & \cdots & a_{1s} \\ \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & \cdots & a_{ss} \\ \hline & b_1 & \cdots & b_s \end{array} \tag{6}$$

lassen sich die Koeffizienten a_{ij}, b_j und c_j eines s-stufigen Verfahrens kompakt darstellen. Runge-Kutta-Verfahren lassen sich in drei verschiedene Verfahrensklassen unterteilen. Ist $\boldsymbol{A}$ eine strikte untere Dreiecksmatrix, so spricht man von einem expliziten Verfahren. Das heißt, dass die Berechnungsvorschriften der Stützstellen nur von bereits berechneten Stützstellen abhängen und so eine sequenzielle

Auswertung möglich ist. Ein semi-explizites Verfahren ergibt sich, wenn $\boldsymbol{A}$ ein Dreiecksmatrix ist. Für jede andere Form der Matrix $\boldsymbol{A}$ handelt es sich um ein implizites Verfahren. Hierbei kann Gleichung (5a) nicht getrennt für die einzelnen Stützstellen betrachtet werden, was zur Folge hat, dass zu jedem Zeitschritt ein nichtlineares Gleichungssystem gelöst werden muss. Für die Lösung differential-algebraischer Gleichungen eignen sich nun besonders die zuletzt genannten impliziten Verfahren.

3 Implizite Runge-Kutta-Verfahren

Eine Vielzahl an implizite Verfahren lassen sich als Kollokationsverfahren auffassen. Der Grundgedanke dieser Verfahren ist die stückweise Approximation der Lösung im Intervall $t \in [t_k, t_{k+1}]$ durch Polynome, die an den Stützstellen die Differentialgleichung bzw. differential-algebraischen Gleichung erfüllen. Zur Bestimmung des Butcher-Blocks wird von verschobenen Legendre-Polynomen ausgegangen:

$$\frac{\mathrm{d}^s}{\mathrm{d}x^s}\left(x^s(x-1)^s\right) \qquad \text{(Gauß-Verfahren)} \tag{7a}$$

$$\frac{\mathrm{d}^{s-1}}{\mathrm{d}x^{s-1}}\left(x^s(x-1)^{s-1}\right) \qquad \text{(RadauIA-Verfahren)} \tag{7b}$$

$$\frac{\mathrm{d}^{s-1}}{\mathrm{d}x^{s-1}}\left(x^{s-1}(x-1)^s\right) \qquad \text{(RadauIIA-Verfahren)} \tag{7c}$$

$$\frac{\mathrm{d}^{s-2}}{\mathrm{d}x^{s-2}}\left(x^{s-1}(x-1)^{s-1}\right) \qquad \text{(Lobatto-Verfahren)} \tag{7d}$$

Diese Polynome zeichnen sich dadurch aus, dass alle Nullstellen im abgeschlossenen Intervall $[0, 1]$ liegen. Diese Nullstellen ergeben die s Koeffizienten c_i, auf deren Basis die Koeffizienten a_{ij} und b_i berechnet werden können.

Im weiteren Verlauf werden anhand des dreistufigen *RadauIIA*-Verfahren[1] die einzelnen Schritte zur Lösung differential-algebraischer Gleichungen aufgezeigt. Dieses Verfahren bietet sich aufgrund seiner Struktur besonders für die Lösung steifer und differential-algebraischer Gleichungen an. Drei Eigenschaften sollten dabei Erwähnung finden:

1. Es gilt $c_s = 1$ und somit unterliegen die algebraischen Nebenbedingungen zum Zeitpunkt t_{k+1} der Fehlerkontrolle. Das heißt, dass beim nächsten Zeitschritt konsistente Anfangsbedingungen vorliegen.

2. Die Matrix $\boldsymbol{A}$ vollen Rang und ist somit invertierbar.

3. Der Vektor $\boldsymbol{b}$ entspricht der letzten Zeile der Matrix $\boldsymbol{A}$, wodurch der neue Zustandsvektor der letzten Stützstelle entspricht und somit weitere Funktionsauswertungen entfallen.

Wie sich später zeigt, ermöglichen die beiden letzten Eigenschaften eine sowohl gut konditionierte als auch effiziente Berechnung.

Ausgehend von Gleichung (5) wird die neue Größe $\boldsymbol{z}_i = \boldsymbol{g}_i - \boldsymbol{y}_k,\ i = 1, \ldots, s$ eingeführt. Damit lässt sich Gleichung (5a) zu

$$\boldsymbol{z}_i = h \sum_{j=1}^{s} a_{ij} \boldsymbol{f}(t_k + c_j h, \boldsymbol{y}_k + \boldsymbol{z}_j) \tag{8}$$

umformulieren. In Matrixschreibweise lautet diese Gleichung

$$\begin{bmatrix} \boldsymbol{z}_1 \\ \vdots \\ \boldsymbol{z}_s \end{bmatrix} = \boldsymbol{A} \otimes \boldsymbol{E}_n \begin{bmatrix} h\boldsymbol{f}(t_n + c_1 h, \boldsymbol{y}_k + \boldsymbol{z}_1) \\ \vdots \\ h\boldsymbol{f}(t_k + c_s h, \boldsymbol{y}_k + \boldsymbol{z}_s) \end{bmatrix}, \tag{9}$$

[1] Der entsprechende Butcher-Block ist im Anhang dargestellt.

wobei $\boldsymbol{E}_n$ eine Einheitsmatrix der Größe n ist und $\otimes$ das Kronecker-Produkt[2] vermittelt. Stellt man das Gleichungssystem nach dem rechten Vektor um und setzt diesen in Gleichung (5b) ein, ergibt sich

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \boldsymbol{b}\boldsymbol{A}^{-1} \otimes \boldsymbol{E}_n \begin{bmatrix} \boldsymbol{z}_1 \\ \vdots \\ \boldsymbol{z}_s \end{bmatrix} \tag{10}$$

Da der Vektor $\boldsymbol{b}$ der letzten Zeile der Matrix $\boldsymbol{A}$ entspricht, ergibt sich

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \boldsymbol{z}_s \ . \tag{11}$$

Damit ergibt sich die Berechnungsvorschrift des Zustandsvektors $\boldsymbol{y}_{k+1}$ in Abhängigkeit der letzten Stützstelle $\boldsymbol{z}_s$ und des Startwerts $\boldsymbol{y}_k$.

Beispiel: Euler rückwärts

Das implizite Euler-Verfahren ist ein einstufiges Runge-Kutta-Verfahren, dass durch den Butcher-Block

$$\begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array} \tag{12}$$

beschrieben wird. Die Berechnungsvorschrift ergibt sich demnach zu

$$\boldsymbol{g}_1 = \boldsymbol{y}_k + h\boldsymbol{f}(t_k + h, \boldsymbol{g}_1) \tag{13a}$$
$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + h\boldsymbol{f}(t_k + h, \boldsymbol{g}_1) \tag{13b}$$

bzw. mit den hier möglichen Vereinfachungen $\boldsymbol{g}_1 = \boldsymbol{y}_{k+1}$ und $t_k + h = t_{k+1}$ zu

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + h\boldsymbol{f}(t_{k+1}, \boldsymbol{y}_{k+1}) \ . \tag{14}$$

Dieses Verfahren nähert die Lösung der Differentialgleichung durch ein Polynom

$$\boldsymbol{p}(t) \text{ für } t \in [t_k, t_{k+1}] \tag{15}$$

ersten Grades an. Dabei entspricht die erste zeitliche Ableitung des Polynoms der an der Stützstelle ausgewerteten Differentialgleichung.

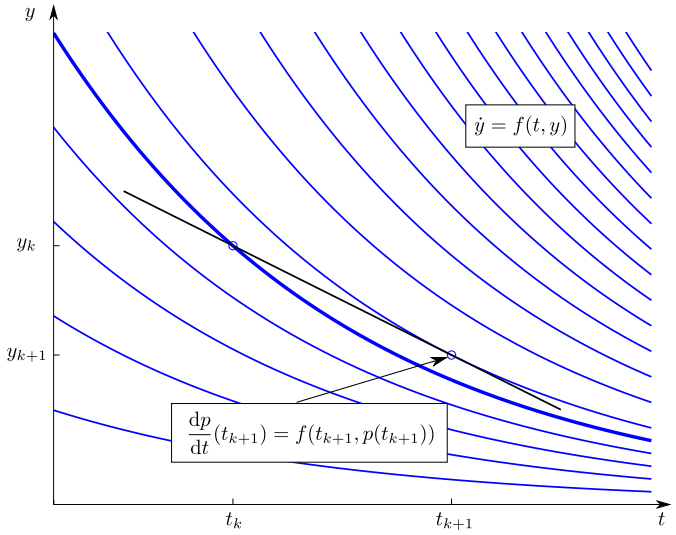

Abbildung 1: Grafische Interpretation des Euler-rückwärts-Verfahrens.

[2] s. Anhang

Abbildung 1 zeigt die grundlegenden Eigenschaften dieses Verfahrens. Die Steigung der Gerade entspricht der an der Stützstelle ausgewerteten Differentialgleichung. Um diese Stützstelle zu finden, bedient man sich eines vereinfachten Newton-Verfahrens.

4 Vereinfachtes Newton-Verfahren

Um Gleichung (9) nach den Unbekannten $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_s$ zu lösen, wird die Gleichung zunächst in ihr implizites Äquivalent gemäß

$$\begin{bmatrix} \boldsymbol{z}_1 \\ \vdots \\ \boldsymbol{z}_s \end{bmatrix} - \boldsymbol{A} \otimes \boldsymbol{E}_n \begin{bmatrix} h\boldsymbol{f}(t_n + c_1 h, \boldsymbol{y}_n + \boldsymbol{z}_1) \\ \vdots \\ h\boldsymbol{f}(t_k + c_s h, \boldsymbol{y}_k + \boldsymbol{z}_s) \end{bmatrix} = \boldsymbol{0} \tag{16}$$

umgeformt. Zur Lösung dieses nichtlinearen Gleichungssystems wird ein vereinfachtes Newton-Verfahren verwendet. Es wird jedoch statt der analytischen Jacobi-Matrix die approximierte Jacobi-Matrix

$$\boldsymbol{J}_{global} = \boldsymbol{E} - h\boldsymbol{A} \otimes \boldsymbol{J}_{approx} \tag{17}$$

herangezogen. Dabei wird $\boldsymbol{J}_{approx} = \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{y}_k}(t_k, \boldsymbol{y}_k)$ anstatt der partiellen Ableitungen $\boldsymbol{J}_{exakt} = \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}_i}(t_k + c_i h, \boldsymbol{y}_k + \boldsymbol{z}_i)$ verwendet. Dadurch wird zwar die quadratische Konvergenz des Newton-Verfahrens aufgegeben, jedoch werden $(s-1)n^2$ Funktionsauswertungen eingespart. Mit Hilfe dieser Jacobi-Matrix kann nun nach den Stützstellen $\boldsymbol{z}_i$ iterativ aufgelöst werden.

Um ein differential-algebraisches Gleichungssystem zu lösen, wird gefordert, dass die Nebenbedingungen an allen Stützstellen gemäß

$$\begin{bmatrix} \boldsymbol{c}(t_k + c_1 h, \boldsymbol{y}_k + \boldsymbol{z}_1, \boldsymbol{u}_k + \boldsymbol{l}_1) \\ \vdots \\ \boldsymbol{c}(t_k + c_s h, \boldsymbol{y}_k + \boldsymbol{z}_s, \boldsymbol{u}_k + \boldsymbol{l}_s) \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{bmatrix} \tag{18}$$

erfüllt werden. Die gesamte rechte Seite ergibt sich damit aus Gl. (16) und(18) zu

$$\begin{bmatrix} \boldsymbol{z}_1 - h \sum_{j=1}^{s} a_{1j} \boldsymbol{f}(t_k + c_j h, \boldsymbol{y}_k + \boldsymbol{z}_j, \boldsymbol{u}_k + \boldsymbol{l}_j) \\ \vdots \\ \boldsymbol{z}_s - h \sum_{j=1}^{s} a_{sj} \boldsymbol{f}(t_k + c_j h, \boldsymbol{y}_k + \boldsymbol{z}_j, \boldsymbol{u}_k + \boldsymbol{l}_j) \\ \boldsymbol{c}(t_k + c_1 h, \boldsymbol{y}_k + \boldsymbol{z}_1, \boldsymbol{u}_k + \boldsymbol{l}_1) \\ \vdots \\ \boldsymbol{c}(t_k + c_s h, \boldsymbol{y}_k + \boldsymbol{z}_s, \boldsymbol{u}_k + \boldsymbol{l}_s) \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{bmatrix}. \tag{19}$$

Die globale Jacobi-Matrix ist dementsprechend auch um die partiellen Ableitungen nach $\boldsymbol{u}$, sowie um die partiellen Ableitungen der Nebenbedingungen gemäß

$$\boldsymbol{J}_{global} = \begin{bmatrix} \boldsymbol{E} - h\boldsymbol{A} \otimes \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{y}_k}(t_k, \boldsymbol{y}_k, \boldsymbol{u}_k) & -h\boldsymbol{A} \otimes \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}_k}(t_k, \boldsymbol{y}_k, \boldsymbol{u}_k) \\ \boldsymbol{E} \otimes \dfrac{\partial \boldsymbol{c}}{\partial \boldsymbol{y}_k}(t_k, \boldsymbol{y}_k, \boldsymbol{u}_k) & \boldsymbol{E} \otimes \dfrac{\partial \boldsymbol{c}}{\partial \boldsymbol{u}_k}(t_k, \boldsymbol{y}_k, \boldsymbol{u}_k) \end{bmatrix} \tag{20}$$

zu erweitern. Das Newton-Verfahren ergibt sich damit zu

$$\boldsymbol{J}_{global} \cdot \begin{bmatrix} \Delta \boldsymbol{z}_1 \\ \vdots \\ \Delta \boldsymbol{z}_s \\ \Delta \boldsymbol{l}_1 \\ \vdots \\ \Delta \boldsymbol{l}_s \end{bmatrix} = - \begin{bmatrix} \boldsymbol{z}_1 - h \sum_{j=1}^{s} a_{1j} \boldsymbol{f}(t_k + c_j h, \boldsymbol{y}_k + \boldsymbol{z}_j, \boldsymbol{u}_k + \boldsymbol{l}_j) \\ \vdots \\ \boldsymbol{z}_s - h \sum_{j=1}^{s} a_{sj} \boldsymbol{f}(t_k + c_j h, \boldsymbol{y}_k + \boldsymbol{z}_j, \boldsymbol{u}_k + \boldsymbol{l}_j) \\ \boldsymbol{c}(t_k + c_1 h, \boldsymbol{y}_k + \boldsymbol{z}_1, \boldsymbol{u}_k + \boldsymbol{l}_1) \\ \vdots \\ \boldsymbol{c}(t_k + c_s h, \boldsymbol{y}_k + \boldsymbol{z}_s, \boldsymbol{u}_k + \boldsymbol{l}_s) \end{bmatrix} . \tag{21}$$

Für die Lösung des linearen Gleichungssystems (21) bietet es sich an, die algebraischen Nebenbedingungen mit der Schrittweite h zu skalieren. Dadurch wird die Pivot-Suche dahingehend beeinflusst, dass die Nebenbedingungen nicht in den oberen Teil des Gleichungssystems wandern. Dies erhöht sowohl die Stabilität als auch die Genauigkeit.

5 Schrittweitensteuerung

Um die Stabilität und Konvergenz des Einschrittverfahrens zu garantieren, muss eine Anpassung der Schrittweite basierend auf einer Fehlerabschätzung erfolgen. Dabei kommt eine Schrittweitensteuerung, die die Lösungen von Verfahren verschiedener Ordnung miteinander vergleicht, zum Einsatz. Die Lösung des eingebetteten Verfahrens ergibt sich aus

$$\hat{\boldsymbol{y}}_{k+1} = \boldsymbol{y}_k + h \left(\hat{b}_0 \boldsymbol{f}(t_k, \boldsymbol{y}_k, \boldsymbol{u}_k) + \sum_{j=1}^{s} \hat{b}_j \boldsymbol{f}(t_k + c_j h, \boldsymbol{y}_k + \boldsymbol{z}_j, \boldsymbol{u}_k + \boldsymbol{l}_j) \right) , \tag{22}$$

wobei der Faktor $\hat{b}_0$ der Kehrwert des reellen Eigenwertes[3] der Matrix $\boldsymbol{A}$ ist. Aus der Differenz der beiden Lösungen folgt

$$\hat{\boldsymbol{y}}_{k+1} - \boldsymbol{y}_{k+1} = \hat{b}_0 h \boldsymbol{f}(t_k, \boldsymbol{y}_k, \boldsymbol{u}_k) + \boldsymbol{e}\boldsymbol{z} . \tag{23}$$

Der Vektor $\boldsymbol{e}$ ergibt sich dabei aus

$$\boldsymbol{e} = \left(\hat{\boldsymbol{b}} - \boldsymbol{b} \right) \boldsymbol{A}^{-1} , \tag{24}$$

wobei der Vektor $\hat{\boldsymbol{b}}$ aus

$$\hat{\boldsymbol{b}} = \begin{bmatrix} 1 - \hat{b}_0 & \frac{1}{2} & \cdots & \frac{1}{s} \end{bmatrix} \begin{bmatrix} 1 & \cdots & c_i^{s-1} \\ \vdots & \ddots & \vdots \\ 1 & \cdots & c_s^{s-1} \end{bmatrix}^{-1} \tag{25}$$

folgt. Da die Differenz $\hat{y}_{k+1} - y_{k+1}$ für $h\lambda \to \infty$ gegen $\hat{b}_0 h \lambda y_k$ strebt und damit unbeschränkt ist, schlug [Shampine80] vor, den Fehler ϵ als

$$\epsilon = (\boldsymbol{E} - h\hat{b}_0 \boldsymbol{J}_{approx})^{-1} (\hat{\boldsymbol{y}}_{k+1} - \boldsymbol{y}_{k+1}) \tag{26}$$

zu definieren. Basierend auf diesem Fehler kann nun eine optimale Schrittweite gefunden werden. Dabei eignen sich vor allem die Fehlberg- ([Fehlberg69]) sowie die Gustafsson-Schrittweitensteuerung ([Gustafsson94]).

[3] Die Matrix $\boldsymbol{A}$ hat bei ungerader Stufenzahl einen reellen Eigenwert und $s-1$ komplexe Eigenwerte

Anhang

Kronecker-Produkt

Das Kronecker-Produkt zweier Matrizen $\boldsymbol{A} \in \mathcal{R}^{m\times n}$ und $\boldsymbol{B} \in \mathcal{R}^{o\times p}$ ist definiert als

$$\boldsymbol{C} = \boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & \cdots & a_{1n}\boldsymbol{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\boldsymbol{B} & \cdots & a_{mn}\boldsymbol{B} \end{bmatrix} , \tag{27}$$

wobei die Matrix $\boldsymbol{C}$ der Dimension $mo \times np$ ist. Falls die Matrizen $\boldsymbol{A}$ und $\boldsymbol{B}$ invertierbar sind, gilt für die inverse Matrix

$$(\boldsymbol{A} \otimes \boldsymbol{B})^{-1} = \boldsymbol{A}^{-1} \otimes \boldsymbol{B}^{-1} . \tag{28}$$

Diese Beziehung gilt ohne Forderung an die Matrizen $\boldsymbol{A}$ und $\boldsymbol{B}$ analog auch für die transponierte, konjugiert komplexe oder adjungierte Matrix.

Das drei-stufige RadauIIA-Verfahren

Eine beliebte Methode aus der Familie der impliziten Runge-Kutta-Verfahren ist das drei-stufige RadauIIA-Verfahren. Diese Methode eignet sich besonders für steife und differential-algebraische Probleme. Der Butcher-Block dieses Verfahrens ergibt sich zu

$$\begin{array}{c|ccc} \frac{4-\sqrt{6}}{10} & \frac{88-7\sqrt{6}}{360} & \frac{296-169\sqrt{6}}{1800} & \frac{-2+3\sqrt{6}}{225} \\ \frac{4+\sqrt{6}}{10} & \frac{296+169\sqrt{6}}{1800} & \frac{88+7\sqrt{6}}{360} & \frac{-2-3\sqrt{6}}{225} \\ 1 & \frac{16-\sqrt{6}}{36} & \frac{16+\sqrt{6}}{36} & \frac{1}{9} \\ \hline & \frac{16-\sqrt{6}}{36} & \frac{16+\sqrt{6}}{36} & \frac{1}{9} \end{array} . \tag{29}$$

Der Vektor $\boldsymbol{b}$ entspricht der letzten Zeile von $\boldsymbol{A}$, was bedeutet, dass die letzte Stützstelle dem neuen Funktionswert entspricht.

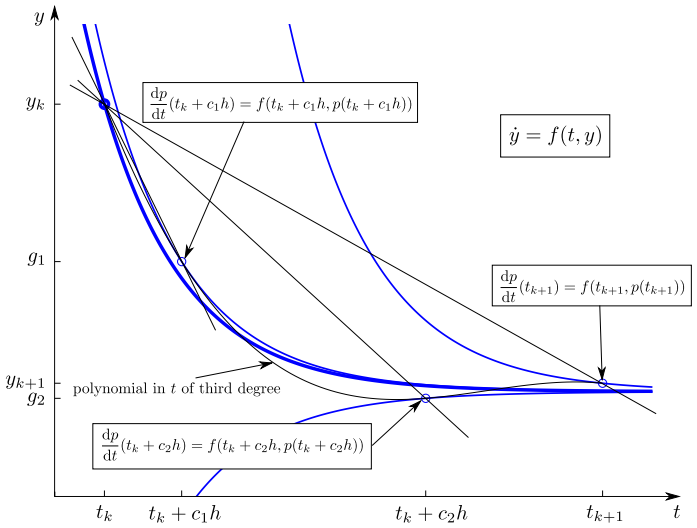

Abbildung 2: Grafische Interpretation des drei-stufigen RadauIIA-Verfahrens mit einer sehr großen Schrittweite h.

Abbildung 2 zeigt die typischen Eigenschaften des RadauIIA-Verfahrens. Die erste zeitliche Ableitung des Polynom stimmt an den drei Stützstellen g_1, g_2 und y_{k+1} mit der zugrunde liegenden Differentialgleichung überein.

Literatur

[Campbell07] Campbell, S.L.: High-index differential algebraic equations. Mechanics of Structures and Machines, Bd. 23, Nr. 2, S. 37–41, 2007.

[Fehlberg69] Fehlberg, E.: Low-order classical Runge-Kutta formulas with step size control and their application to some heat transfer problems. Techn. Bericht. 315, National Aeronautics and Space Administration, 1969.

[FhrerLeimkuhler91] Fhrer, C.; Leimkuhler, B.: Numerical solution of differential-algebraic equations for constrained mechanical motion. Numerische Mathematik, Bd. 59, S. 55–69, 1991.

[Gustafsson94] Gustafsson, K.: Control-theoretic techniques for stepsize selection in implicit Runge-Kutta methods. ACM Trans. Math. Soft., Bd. 20, S. 496–517, 1994.

[HairerLubichRoche89] Hairer, E.; Lubich, C.; Roche, M.: The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods, Bd. 1409 von Lecture Notes in Mathematics. Springer-Verlag, 1989.

[HairerWanner10] Hairer, E.; Wanner, G.: Solving Ordinary Differential Equations II - Stiff and Differential-Algebraic Problems. Berlin: Springer, 2. Aufl., 2010.

[Shampine80] Shampine, L.: Implementation of implicit formulas for the solution of ODEs. SIAM Journal of Stat. Computations, Bd. 2, S. 44–53, 1980.