Leistungs- und Genauigkeitsanalyse numerischer Löser für Differentialgleichungen

Bachelorarbeit

Arbeitsbereich Wissenschaftliches Rechnen
Fachbereich Informatik
Fakultät für Mathematik, Informatik und Naturwissenschaften
Universität Hamburg

Vorgelegt von:	Joel Graef
E-Mail-Adresse:	2graef@informatik.uni-hamburg.de
Matrikelnummer:	6416962
Studiengang:	Computing in Science
Erstgutachter:	Dr. Michael Kuhn
Zweitgutachter:	Prof. Dr. Thomas Ludwig
Betreuer:	Fabian Große, Dr. Michael Kuhn

Hamburg, den 12.09.2016

Kurzfassung

Diese Bachelorarbeit beschäftigt sich mit der Frage, ob Lösungsverfahren für Differentialgleichungen mit höherer Ordnung in jedem Fall besser für die Verwendung in numerischen Modellen geeignet sind als solche mit niedrigerer Ordnung. Die Frage wird unter Verwendung von vier Lösungsverfahren im Hinblick auf zwei verschiedene Differentialgleichungen und ein NPD-Modell (Nährstoff-Phytoplankton-Detritus), welches ein vereinfachtes marines Ökosystem beschreibt, geklärt. Zunächst werden einige Hintergrundaspekte zu Lösungsverfahren für Differentialgleichungen vorgestellt und auf Einschritt- und Mehrschrittverfahren eingegangen. Hierbei werden insbesondere die verwendeten Lösungsverfahren nach Euler, Heun, Adams-Bashforth 2. Ordnung (AB2) und Runge-Kutta 4. Ordnung (RK4) behandelt.

Bei der Leistungsanalyse werden die Verfahren hinsichtlich ihrer Genauigkeit und Laufzeit verglichen. Außerdem wird eine Schrittweitensteuerung vorgestellt, die bei einer Abweichung der Approximation zur analytischen Lösung die Schrittweite reduziert und nach einem bestimmten Intervall wieder erhöht. Sowohl mit Schrittweitensteuerung als auch ohne erreichte das Verfahren höchster Ordnung (RK4) die beste Laufzeit.

Unter Verwendung eines NPD-Modells werden die Verfahren mit Ausnahme des AB2-Verfahren ebenfalls analysiert. Dabei wird festgestellt, dass sich die Nutzung vom Heun-, AB2- und RK4-Verfahren gegenüber dem Euler-Verfahren für das Modell nicht rentieren. Ausschlaggebend dafür ist die Wahl der Schrittweite, die von der Genauigkeit der Verfahren abhängt. Die Genauigkeit wird durch die Berechnung von Zusatzrechenschritten erhöht und erlaubt damit die Wahl eines gröberen Zeitschritts. Die Rechenzeit pro Zusatzrechenschritt ist bei der Nutzung des NPD-Modells größer als die Rechenzeiteinsparung durch den gröberen Zeitschritt. Da aber beispielsweise keine Schrittweitensteuerung im Modell implementiert wurde, bestehen durchaus weitere Ansatzpunkte zur Verbesserung der Laufzeit.

Inhaltsverzeichnis

1 Einleitung

Differentialgleichungen (DGL) sind mathematische Gleichungen die Beziehungen zwischen einer Funktion und mindestens einer Ableitung dieser Funktion beschreiben. Genutzt werden sie unter anderem zur Modellierung von Naturgesetzen wie z. B. der Gezeitenkräfte [Wik16a]. Das Ziel dieser Bachelorarbeit ist es, numerische Lösungsverfahren für DGL bezüglich ihrer Leistung und Genauigkeit zu vergleichen und zu bewerten. Hierbei soll untersucht werden, ob ein Lösungsverfahren mit höherer Genauigkeit und daraus resultierend mit weniger Rechenschritten in jedem Fall schneller ist, als eines mit geringerer Genauigkeit und einer höheren Anzahl an Rechenschritten. Ein Anwendungsfall ist die energieeffiziente Nutzung von Simulationen in Rechenzentren wie dem Deutschen Klimarechenzentrum [Deu16].

Rechnergestützte numerische Modelle sind in den vergangenen Jahrzehnten immer wichtiger geworden. Sie werden zum Beispiel in der Medizin zur Analyse von Gewebebildungen, in Ingenieurwissenschaften für die Materialentwicklung oder in der Meteorologie zur Wettervorhersage verwendet [Wis14]. In vielen Bereichen werden sie vor dem Hintergrund genutzt, dass eine Analyse rein theoretischer oder experimenteller Natur nicht ausreicht oder gar fehlschlägt. Damit die Simulationen möglichst genaue Realitätsabbildungen darstellen, ist der Gebrauch von vielen Parametern - die Einflüsse der Umgebung wie Wachstums- oder Sterberaten darstellen - oder die Berechnung eines möglichst langen Zeitraums zwingend notwendig. Je genauer Simulationen die Realität beschreiben sollen, desto mehr Faktoren erhöhen die Laufzeit. Um eine gewisse Leistung zu gewährleisten und trotzdem realitätsnahe Ergebnisse zu erzielen, müssen unter anderem die Differentialgleichungslöser möglichst effizient sein. Die Wahl des am besten geeigneten Algorithmus für die jeweilige Modellanwendung stellt deshalb einen zentralen Punkt dar.

1.1 Motivation

Gewonnene Daten (wie z. B. Langzeitmessungen) werden dazu verwendet, physikalische, chemische oder biologische Zusammenhänge zu erkennen. Diese werden anschließend in Form von DGLs mathematisch beschrieben und können durch Simulationen analysiert werden. In Fällen in denen keine analytische Lösung bestimmt werden kann, wie z. B. bei den Navier-Stokes-Gleichungen [Wik16f], müssen die DGLs diskretisiert und numerisch gelöst werden. Diese diskretisierten DGLs stellen dann in Kombination mit den numerischen Lösungsverfahren die Basis für die Modelle zur Simulation dar.

Numerische Lösungsverfahren für Differentialgleichungen gibt es viele. Diese können größtenteils in verschiedene Familien unterteilt werden und weisen verschiedene Vor- und

Nachteile auf. Je nach Anwendungsgebiet gibt es Familien von Lösungsverfahren die besser geeignet sind. Auch können durch Anpassungen verschiedener Faktoren aus Grundformeln einer Familie eigene Lösungsverfahren erstellt werden. Aufgrund verschiedener Größenordnungen der lokalen und globalen Fehler der Verfahren (auch Konvergenzordnung genannt) und dadurch auch anderer Faktoren, wie z. B. die Wahl der Schrittweite, ist die Analyse numerischer Lösungsverfahren ein wichtiger Bestandteil der Leistungsverbesserung von Programmen, die Simulationen durchführen. Zwei der bekanntesten Lösungsfamilien werden in dieser Bachelorarbeit im Kapitel *Hintergrund* vorgestellt.

1.2 Lösungsansatz

Als Grundlage für die Analysen wurden vom Autor vier Lösungsalgorithmen implementiert und unter verschiedenen Bedingungen verglichen. Die Algorithmen, bei denen es sich um das Euler-Verfahren, das Runge-Kutta-Verfahren vierter Ordnung (RK4), das Heun-Verfahren und das Adams-Bashforth-Verfahren zweiter Ordnung (AB2) handelt, wurden mit analytischen Lösungen verglichen und grafische Auftragungen erstellt. Anschließend wurden das Heun- und das RK4-Verfahren ebenfalls in einem weiteren Programm implementiert, welches von Johannes Pätsch (Arbeitsbereich *Theoretische Ozeanographie*, Institut für Meereskunde, Uni Hamburg) geschrieben wurde. Dieses Programm stellt ein NPD-Modell dar (Nährstoff-Phytoplankton-Detritus) und beschreibt die zeitliche Entwicklung dieser drei Zustandsvariablen durch Differentialgleichungen. Diese werden unter Verwendung des Euler-Verfahrens gelöst. Bei dem NPD-Modell handelt es sich um die Abbildung eines einfachen Stoffkreislaufs, bei dem ein Element wie z. B. Stickstoff verschiedene anorganische (Nährstoffe) und organische Zustände (Phytoplankton = Algen, Detritus = totes, organisches Material) durchläuft. Näheres zum Modell ist im Kapitel *Verwendungszweck* nachzulesen.

1.3 Struktur der Arbeit

Zuerst wird im Kapitel *Hintergrund* ein Einblick in Differentialgleichungen und numerische Lösungsverfahren gegeben. Dabei wird auf Einschritt- und Mehrschrittverfahren eingegangen und die Familien der Runge-Kutta-, sowie der Adams-Bashforth-Verfahren werden vorgestellt. Danach werden in dem Kapitel *Programmbeschreibung* der Verwendungszweck und die geschriebenen Analyseprogramme aufgezeigt. Im Anschluss daran werden im Kapitel *Durchführung und Auswertung* die Ergebnisse gezeigt, veranschaulicht und eine Bewertung der Ergebnisse gegeben. Zum Schluss wird ein Fazit gezogen und auf Möglichkeiten zur weiteren Analyse eingegangen.

2 Hintergrund

In diesem Kapitel wird eine Einführung in die Familie der gewöhnlichen Differentialgleichungen und in ihre Lösungsalgorithmen gegeben. Dabei wird auf Einschritt- und Mehrschrittverfahren eingegangen und einige Verfahren vorgestellt.

2.1 Gewöhnliche Differentialgleichungen

Bei gewöhnlichen Differentialgleichungen (DGL) handelt es sich um Gleichungen für gesuchte Funktionen, die die Ableitung von genau einer Variable beinhalten. Diese stellen mathematisch beschriebene physikalische, chemische oder biologische Zusammenhänge dar und ermöglichen die Analyse eben dieser.

Bei den DGL wird zwischen den expliziten und den impliziten unterschieden. Die expliziten DGL sind im Allgemeinen wie folgt aufgebaut

$$y^{(n)} = F(x, y, y', y'', \cdots, y^{(n-1)}) \tag{2.1}$$

und werden durch die höchste vorkommende Ableitung definiert. Dabei bezeichnet F die Gleichung und $y^{(n)}$ die n-te Ableitung nach der Unbekannten x. Implizite DGL sind dagegen definiert durch

$$F(x, y, y', y'', \cdots, y^{(n)}) = 0 \tag{2.2}$$

In beiden Fällen bestimmt die höchste Ableitung die Ordnung der DGL. Wie in Gleichung (2.1) beschrieben, wird eine DGL, die nach der höchsten Ableitung umgeformt werden kann, als explizit bezeichnet [Win16].

Nur wenige Typen von DGLs lassen sich näherungsweise lösen; zu diesen zählen die expliziten gewöhnlichen DGLs. Diese Lösungsverfahren bauen im Grunde alle auf dem Anfangswertproblem auf, bei dem die Lösung der DGL unter Berücksichtigung eines Anfangswertes ermittelt wird. Außerdem wird das zu berechnende Intervall in endlich viele kleinere Intervalle aufgeteilt. In Abbildung 2.1 wird dieses Grundprinzip numerischer Lösungsverfahren für DGLs beschrieben. Dabei wird anstelle der kontinuierlichen analytischen Lösung $y(t)$ (blau) im Intervall $[t_0; t_b]$ die Zerlegung in kleinere Teilintervalle ($[t_0; t_1], [t_1; t_2], \cdots, [t_{m-1}; t_m]$ mit $t_m = t_b$) betrachtet und diese approximiert (rot). Diese Approximation $Y_j \approx y(t_j)$ (mit j = 0,1, ... m) erfolgt über den Differentialquotienten, also die Änderungsrate in Abhängigkeit zur Zeit. Der Differentialquotient wird zur Entwicklung von Verfahren zur schrittweisen Annäherung der Lösung verwendet und stellt ihren Grundbaustein dar. Diese Lösungsverfahren können in drei Klassen aufgeteilt werden: in die Einschritt-, Mehrschritt- und Extrapolationsverfahren. Im Folgenden wird nur auf die ersten beiden Klassen eingegangen, da aus diesen die am meisten verwendeten Verfahren stammen.

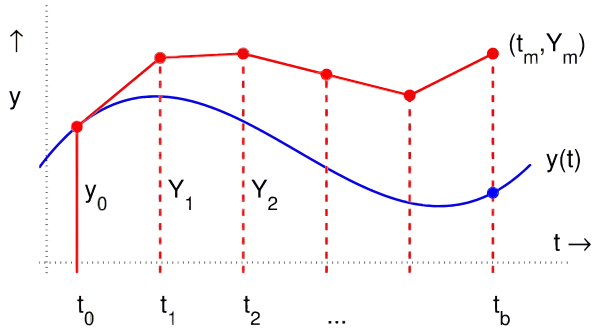


Abbildung 2.1: Prinzip der Lösungsalgorithmen. Analytische Lösung (blau) und Näherung (rot). [Obe08, S. 51]

2.2 Einschritt- und Mehrschrittverfahren

Bei den Einschrittverfahren wird zur Bestimmung des nächsten Punktes (t_{i+1}, y_{i+1}) immer die vorherige Näherung (t_i, y_i) mit einbezogen, wobei t den Zeitpunkt, y den Funktionswert und i den Schritt bezeichnet. Die Familie der Runge-Kutta-Verfahren ist eine der wichtigsten Vertreterinnen dieser Klasse. Alle Verfahren der Familie bauen auf der gleichen Grundformel auf (siehe Gleichung (2.3)). In der Regel ist allerdings von dem Runge-Kutta-Verfahren vierter Ordnung (RK4) die Rede (siehe Gleichung (2.4)).

$$y_{n+1} = y_n + h \sum_{j=1}^{s} b_j k_j \tag{2.3}$$

$$y_{n+1} = y_n + h \cdot (\frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4) \tag{2.4}$$

Die allgemeine Formel besteht aus den Koeffizienten b_j, die das jeweilige Verfahren definieren, den k_j Zwischenschritten zur Genauigkeitserhöhung, der Länge der Teilintervalle bzw. der Schrittweite h und der Ordnung des Verfahrens s. Die Zwischenschritte werden mittels der Gleichung (2.5) berechnet.

$$k_j = f(t_n + hc_j, y_n + \sum_{l=1}^{s} a_{jl} k_l), \;\; j = 1, ..., s \tag{2.5}$$

Die a_{jl} und c_j sind dabei weitere Koeffizienten, die je nach Anwendung explizit bestimmt werden können. Das Runge-Kutta-Verfahren erster Ordnung ist hierbei durch $c_1 = 0$ definiert und besitzt keinen a_{jl} Wert, da kein Zwischenschritt k_j berechnet werden muss. Angegeben werden die Koeffizienten b_j, a_{jl} und c_j in der Regel mit dem sogenannten Butcher-Schema bzw. -Tableau (siehe Abb. 2.2) [But08].

$$
\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & & \ddots & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s
\end{array}
$$

Abbildung 2.2: Butcher-Schema zur Angabe der Koeffizienten von Runge-Kutta-Verfahren.

Im Falle des RK4-Verfahrens lauten die verschiedenen Koeffizienten k_j somit:

$$
\begin{aligned}
k_1 &= f(t_n, y_n) \\
k_2 &= f(t_n + \frac{h}{2}, y_n + \frac{h}{2} \cdot k_1) \\
k_3 &= f(t_n + \frac{h}{2}, y_n + \frac{h}{2} \cdot k_2) \\
k_4 &= f(t_n + h, y_n + h \cdot k_3)
\end{aligned}
\tag{2.6}
$$

Das Runge-Kutta-Verfahren erster Ordnung ist auch als Euler-Verfahren (siehe Gleichung (2.7)) bekannt. Dieses benötigt keine Zwischenschritte und ist im Vergleich zu den Verfahren höherer Ordnung bei gleicher Schrittweite zwar schneller, aber deutlich ungenauer. Das Heun-Verfahren (siehe Gleichung (2.8)), bei dem es sich ebenfalls um ein Einschrittverfahren handelt, baut auf dem Euler-Verfahren auf und modifiziert dieses. Durch das Berechnen eines Mittelpunktes von zwei Approximationen mit unterschiedlicher Steigung wird der globale Fehler von $O(h)$ auf $O(h^2)$ verringert [Wik16d, Ber16]; dabei ist h die Schrittweite und in der Regel kleiner eins.

$$
y_{n+1} = y_n + h \cdot f(t_n, y_n)) \tag{2.7}
$$

$$
\begin{aligned}
k &= y_n + h \cdot f(t_n, y_n) \\
y_{n+1} &= y_n + \frac{h}{2}(f(t_n, y_n) + f(t_n + h, k))
\end{aligned}
\tag{2.8}
$$

Mehrschrittverfahren bauen auf mehr als einem vergangenen Schritt auf. Dadurch wird versucht den bisherigen Verlauf der Funktion in die neuen Punkte mit einfließen zu lassen und so eine genauere Lösung zu erzielen. Vertreter dieser Klasse sind unter anderem aus der Familie der Adams-Bashforth-Verfahren. Bei diesen Verfahren wird jeweils ein Einzelschrittverfahren zu Beginn verwendet, um Anfangspunkte für die nachfolgende Berechnung zu erhalten. Da die Familie der Runge-Kutta-Verfahren weit verbreitet ist, werden sie häufig hierfür verwendet. Das Adams-Bashforth-Verfahren zweiter Ordnung erfordert z. B. zuerst die Berechnung eines ersten Punktes durch ein anderes Verfahren,

damit anschließend unter Hinzunahme dieses Punktes der Nächste berechnet werden kann. Für die weiteren Schritte in den Adams-Bashforth-Verfahren werden Gleichungen der Form (2.9) aufgestellt [Wik16e] [Pla00, S. 161-163]:

$$y_{n+s} = y_{n+s-1} + h \cdot (b_s \cdot f(t_{n+s}, y_{n+s}) + b_{s-1} \cdot f(t_{n+s-1}, y_{n+s-1}) + \cdots + b_0 \cdot f(t_n, y_n)) \quad (2.9)$$

Die $b_0, \ldots, b_s$ sind Koeffizienten, die je nach Ordnung s bestimmt werden müssen. Mithilfe der Gleichungen (2.10) und (2.11) kann ein Polynom $p(t)$ des Grades $s-1$ bestimmt werden, das die Differentialgleichung $y' = f(t, y)$ mit $y' = p(t)$ approximiert und in jedem Fall exakt lösbar ist.

$$p(t_{n+i}) = f(t_{n+i}, y_{n+i}), \quad \text{mit } i = 0, \cdots, s-1 \quad (2.10)$$

$$p(t) = \sum_{j=0}^{s-1} \frac{(-1)^{s-j-1} f(t_{n+j}, y_{n+j})}{j!(s-j-1)!h^{s-1}} \prod_{\substack{i=0 \\ i \neq j}}^{s-1} (t - t_{n+1}) \quad (2.11)$$

Die Lösung der entstehenden DGL lässt sich durch die Integration von p bestimmen:

$$y_{n+s} = y_{n+s-1} + \int_{t_{n+s-1}}^{t_{n+s}} p(t)\ dt \quad (2.12)$$

Durch Substitution der Funktion $p(t)$ von Gleichung (2.12) mit Gleichung (2.10) erhält man das jeweilige Adams-Bashforth-Verfahren. Die Koeffizienten b_j sind durch folgende Gleichung gegeben:

$$b_{s-j-1} = \frac{(-1)^j}{j!(s-j-1)!} \int_0^1 \prod_{\substack{i=0 \\ i \neq j}}^{s-1} (u+i)\ du, \quad \text{mit } j = 0, \cdots, s-1 \quad (2.13)$$

Das Adams-Bashforth-Verfahren zweiter Ordnung ist damit durch folgende Formel definiert und benötigt zur Durchführung einen ersten Wert y_{n-1}:

$$y_{n+1} = y_n + h \cdot (\frac{3}{2} f(t_n, y_n) - \frac{1}{2} f(t_{n-1}, y_{n-1})) \quad (2.14)$$

Sowohl Einschritt- als auch Mehrschrittverfahren höherer Ordnung sind langsamer als Verfahren geringerer Ordnung. Da Verfahren höherer Ordnung allerdings in der Regel auch eine höhere Genauigkeit erzielen, kann die genutzte Schrittweite für diese Verfahren höher gewählt werden. In Folge dessen müssen Verfahren mit einer höheren Schrittweite weniger Punkte berechnen und können im Schnitt eine Lösung schneller berechnen als Verfahren mit geringerer Schrittweite. Auch ist es möglich die Schrittweite adaptiv zu wählen. Dies kann zum Beispiel durch eine vordefinierte maximale Abweichung geschehen oder durch einen Wert der in der Realität nicht möglich ist, wie z. B. eine Stoffkonzentration unter Null. Die Berechnung der Abweichung kann beispielsweise durch zweifache Approximation stattfinden, bei der die erste mit ganzer und die zweite mit halber Schrittweite berechnet wird. Da dies aber rechenintensiv ist, wurde z. B. das Runge-Kutta-Fehlberg Verfahren

entwickelt. Bei diesem werden in jedem Schritt zwei unterschiedliche Approximationen berechnet und verglichen. Wenn die Approximationen zu weit voneinander entfernt sind, wird die Schrittweite halbiert und wenn sie nahe aneinander liegen wird eine der Lösungen akzeptiert. Im Fall, dass in den Approximationen mehr als die geforderten signifikanten Stellen übereinstimmen, wird die Schrittweite erhöht [MF04, S. 497-498]. Durch dieses Prinzip kann in den meisten Fällen eine Genauigkeit der Ergebnisse sicher gestellt werden. Weitere Schrittweitenunterteilungen werden im Buch *Numerik gewöhnlicher Differentialgleichungen: Anfangs- und Randwertprobleme* [Her04, S. 65-71] beschrieben.

Nachteil der Mehrschrittverfahren ist, dass durch das Speichern vorheriger Punkte, die zur Berechnung neuer Punkte benötigt werden, auch Ungenauigkeiten in die neuen Punkte übertragen werden. Aus kleinen häufigen Fehlern wie Ungenauigkeiten des Verfahrens können so immer größere Fehler entstehen. Auch ist eine adaptive Schrittweitensteuerung komplizierter als bei den Einschrittverfahren. Vorteil dagegen ist, dass auch bei höherer Ordnung nur eine Funktionsauswertung von f benötigt wird, während bei den Einschrittverfahren immer mehr Zwischenschritte berechnet werden müssen, je höher die Ordnung wird [Vos14, S. 31, 33].

Zusammenfassung

Es gibt verschiedene Klassen von Lösungsalgorithmen für Differentialgleichungen, von denen die Einschritt- und Mehrschrittverfahren vorgestellt wurden. Durch einen Anfangswert und eine Intervallunterteilung kann die Lösung einer DGL unter Verwendung dieser Verfahren approximiert werden. Ein bekanntes Beispiel für die Klasse der Einschrittverfahren ist hierbei die Familie der Runge-Kutta-Verfahren und für die Mehrschrittverfahren die Adams-Bashforth-Familie. Mithilfe von Koeffizienten können die Verfahren beider Familien bei Bedarf angepasst und unterschiedliche Genauigkeiten erzielt werden. Um die Vorzüge der höheren Genauigkeit bei größerer Ordnung zu nutzen, kann die Schrittweite höher, als bei Verfahren mit kleinerer Ordnung, gewählt werden. Mittels einer adaptiven Schrittweitensteuerung können auch zu große Abweichungen von der analytischen Lösung während der Durchführung erkannt und umgangen werden.

3 Programmbeschreibung

Die Beschreibung von natürlichen und naturwissenschaftlichen Prozessen - wie die Zunahme und Abnahme der Temperatur von Objekten, Konzentrationen, Radioaktivität von Stoffen und das Wachstum von Populationen - mithilfe von DGL ist weit verbreitet. Die DGLs werden durch numerische Modelle approximiert und analysiert. Im Folgenden werden Programme zum Vergleich und zur Analyse einiger numerischer Modelle beschrieben, die im Rahmen dieser Bachelorarbeit geschrieben wurden.

3.1 Verwendungszweck

Verwendet werden sollen die Ergebnisse dieser Bachelorarbeit in dem marinen Ökosystemmodell *ECOHAM* [PK08, LPMK12, GGK$^+$16] der Universität Hamburg. Für die Analysen wurde aus Gründen der Übersichtlichkeit ein kleines Programm genutzt, das ein NPD-Modell darstellt (siehe Kapitel *3.3 Programmablauf des NPD-Modells*). Bei einem NPD-Modell handelt es sich um die simpelste Variante eines marinen Ökosystemmodells, in dem Beziehungen zwischen den Nährstoffen N, Phytoplankton P (= Algen) und Detritus D (= totes, organisches Material) dargestellt werden (siehe Abb. 3.1). Im Modell verwendet Phytoplankton das Licht und (anorganische) Nährstoffe (Aufnahme), um Biomasse (= lebendes, organisches Material) aufzubauen. Die anorganischen Nährstoffe werden bei der Zersetzung/Remineralisierung des Detritus freigesetzt. Phytoplankton kann absterben und geht dabei in Detritus über. So ist ein einfacher Kreislauf vorhanden, der den Grundbaustein des marinen Ökosystems darstellt. In den meisten Fällen wird dieses Modell um Zooplankton erweitert (NPZD-Modell), was in einem NPD-Modell allerdings in dem Faktor der Mortalität des Phytoplanktons berücksichtigt werden kann. Näheres zu NPZD-Modellen lässt sich beispielsweise im Buch *Introduction to the Modelling of Marine Ecosystems* finden [FN14, S. 20-22].

Um sich der Fragestellung, ob Lösungsverfahren für DGLs mit höherer Ordnung in jedem Fall besser für die Verwendung in numerischen Modellen geeignet sind als solche mit niedrigerer Ordnung, anzunähern, wurden zwei unterschiedliche Analysen durchgeführt. Zur grundlegenden Analyse der DGL-Lösungsverfahren wurden zuerst vier der Verfahren implementiert (siehe Kapitel *3.2 Programmablauf der Leistungs- und Genauigkeitsanalyse*) und miteinander verglichen (siehe Kapitel *4.2.2 Laufzeit- und Genauigkeitsvergleich*). Für einen tiefergehenden Vergleich der Lösungsverfahren in einer praxisorientierten Modellanwendung ohne lösbare DGL wurde anschließend das oben beschriebene NPD-Modell genutzt.

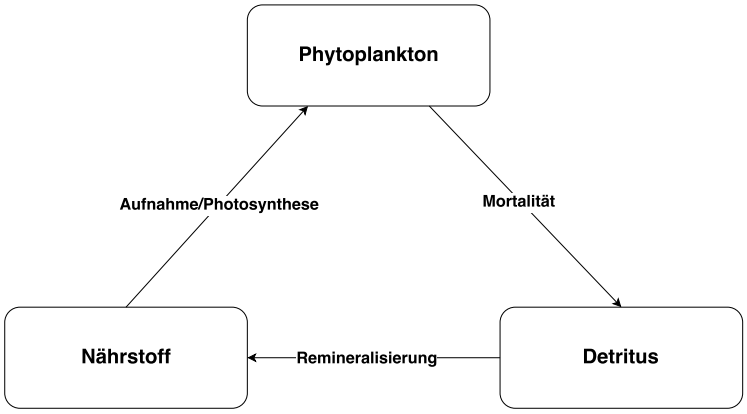


Abbildung 3.1: Interaktionsdiagramm der Zustandsvariablen in einem NPD-Modell. Boxen zeigen die verschiedenen Zustandsvariablen: Nährstoff (N), Phytoplankton (P) und Detritus (D). Pfeile bezeichnen die verschiedenen Prozesse, über welche die Zustandsvariablen miteinander interagieren.

3.2 Programmablauf der Leistungs- und Genauigkeitsanalyse

Die Leistungs- und Genauigkeitsanalyse wurde mittels mehrerer kleiner Programme bestimmt, die jeweils eine Setup-Datei einlesen. In der Datei stehen der x-Startwert `x_start` , eine Zeitschrittweite `delta_x` , im Folgenden auch Schrittweite genannt, und eine Anzahl der zu berechnenden Punkte `nLocs_x` , oder bei einem festen Intervall, ein Endpunkt `x_end` . Implementiert wurden die Lösungsverfahren nach Euler, Heun, AB2 und RK4.

In einem ersten Programm `SolverComparison` werden der Startpunkt, die Schrittweite und die Anzahl der Schritte eingelesen und verwendet. Der Zweck des Programms ist es, mittels erster Analysen des Zeitaufwands und der Genauigkeit Vergleiche zwischen den numerischen Lösungsverfahren zu erstellen.

Zu Beginn wird mit dem Startwert von x ein Anfangswert $y = f(x)$ bestimmt, der die Lösung durch die Verfahren ermöglicht. Anschließend werden durch Iteration über die Anzahl der Punkte in Abhängigkeit zur Schrittweite die exakten Lösungswerte berechnet. Durch den Anfangswert von y erfolgt darauf die Approximation mit den implementierten Lösungsverfahren. Abschließend werden alle berechneten Punkte in eine Datei geschrieben.

Der Zweck eines zweiten Programms (`OptimalStep`) ist die Bestimmung der Schrittweite, bei der die Genauigkeit jedes Verfahrens immer unter einer Abweichung von einem Prozent liegt. Hierfür muss ein `x_start` , sowie ein `nLocs_x` angegeben werden. Da eine optimale Schrittweite bestimmt werden soll, ist die Angabe einer Schrittweite nicht erforderlich. Zu Anfang wird ein hoher Zeitschritt für jedes Verfahren definiert. Dann folgt die Berechnung der exakten und der approximierten Ergebnisse für jeden

Punkt von `nLocs_x` . Im Anschluss daran wird die Abweichung der approximativen Punkte $y(x_i)$ von den exakten ($solution\_analytical(i)$) bestimmt (siehe Listing 3.1). Im Falle, dass die Abweichung größer als ein Prozent ist, wird ein Wahrheitswert auf *true* gesetzt. Anschließend erfolgt eine Halbierung des Schrittes und eine erneute Berechnung der Punkte mit dem neuen Schritt solange bis die Abweichung von der analytischen Lösung kleiner oder gleich 1% ist. Wenn die Wahrheitsvariable nicht gesetzt wird, also *false* bleibt, wird der berechnete Zeitschritt `step1` ausgegeben. Für die Bestimmung der Schrittweite ist dabei die Anzahl der zu berechnenden Punkte signifikant.

```
1     a = (y(i) - solution_analytical(i))
2     b = (solution_analytical(i)/100)
3     if ( (abs(a) > abs(b)) )then
4         valid1 = .true.
5         exit
6     end if
7 end do
8 if (valid1 .eqv. .true.) then
9     step1 = step1/2
10 else
11    exit
12 end if
```

Listing 3.1: Programmabschnitt aus `OptimalStep` zur Bestimmung der optimalen Schrittweite.

Zur Nutzung der durch `OptimalStep` bestimmten optimalen Schrittweite wurde das `SolverComparison` Programm angepasst und separat als `OptimalResults` gespeichert. Anpassungen mussten bei der Übergabe der Schrittweiten und der Abspeicherung in eine Datei vorgenommen werden. Die für jedes Verfahren unterschiedlichen Schrittweiten wurden als neue Parameter anstatt dem allgemeinen `delta_x` eingeführt. Da die Anzahl der Schritte durch die unterschiedlichen Schrittweiten ebenfalls unterschiedlich sind, werden für jedes Verfahren neue `nLocs_x` berechnet. Durch diese kann die Abspeicherung der x- und y-Werte beschränkt werden, sodass zu jedem Verfahren eine Auftragung mit eigener Schrittweite generiert werden kann.

Da eine optimale Schrittweite nur im Hinblick auf die gewünschte Abweichung bestimmt werden kann, dies aber mit einem erheblichen Kostenaufwand verbunden ist, wurde das Prinzip der adaptiven Schrittweitenhalbierung implementiert (siehe Programm `DynStep`). Dieses Prinzip basiert auf der Teilung der Schrittweite, wenn die approximative Bestimmung nicht innerhalb der erlaubten Abweichung liegt.

Im Listing 3.2 wird die implementierte adaptive Schrittweitenhalbierung aufgezeigt, die durch eine rekursive Subroutine unter Nutzung konstanten Speicherbedarfs geschrieben wurde und an die Implementierung dieser im ECOHAM-Modell angelehnt ist. Dabei wird unabhängig von der Größe des zu berechnenden Intervalls Speicherplatz reserviert

und dieser in jeder Iteration mit den neu berechneten Werten überschrieben. Die Routine wird solange aufgerufen, bis das Ende des aktuellen Zeitschritts der Länge `delta_x` erreicht ist. Übergeben werden der Subroutine die aktuelle Schrittweite `step` und ein `level`, das die Anzahl der Schritthalbierungen zählt. In Gleichung (3.1) sieht man den Zusammenhang zwischen den beiden Variablen:

$$\texttt{step} = \texttt{delta\_x} \cdot 0.5^{\texttt{level}} \tag{3.1}$$

Die Abweichung der approximierten zur exakten Berechnung wird analysiert. Das Kriterium ist dabei so gewählt, dass von jedem neu berechneten Punkt die analytisch berechnete Lösung subtrahiert und mit einem Prozent dieser verglichen wird. Wird das Kriterium verletzt, wird die Schrittweite und damit das Intervall halbiert. Anschließend wird die Berechnung auf beiden Hälften des Intervalls mittels einer Rekursion solange fortgeführt wie die Abweichung größer als gewünscht ist. Dabei wird zunächst das `level` um 1 erhöht. Überschreitet das neue level einen Wert von 20, so wird die Rekursion abgebrochen. Dies wird gemacht damit die Rekursion nicht endlos ausgeführt wird, sollte die Approximation zu weit von der analytischen Lösung entfernt sein. In diesem Fall wird der x-Wert auf das Ende des Intervalls gesetzt, eine Abbruch-Variable `abort1` wird aktiviert und durch *return* der Aufruf der laufenden Rekursion beendet. Andernfalls wird die aktuelle Schrittweite erneut halbiert und ein counter zum Zählen der insgesamt benötigten Halbierungen um 1 erhöht. Anschließend wird die Berechnung der Approximation mit der neuen aktuellen Schrittweite fortgesetzt. Dabei wird die rekursive Subroutine aufgrund der Halbierung von step zweimal hintereinander ausgeführt. Durch dieses Kriterium erzielen alle numerischen Lösungsverfahren ähnlich genaue Ergebnisse und können so unter den gleichen Bedingungen verglichen werden.

Zur Analyse der Änderungen von Laufzeiten, dem maximalen Fehler und der Halbierungsanzahlen werden die Verfahren mehrmals mit unterschiedlicher Schrittweite oder Intervallgröße aufgerufen. Die Ergebnisse werden zusammen mit der Schrittweite oder der Intervallgröße in einer Datei ausgegeben. Schlägt ein Verfahren fehl und muss abgebrochen werden, wird für dieses die Laufzeit, Halbierungsanzahl und der Fehler mit *NaN* (Not a Number) beschrieben. Bei der Auftragung werden diese Werte von MATLAB ausgelassen.

```
1  if (abs(y(2) - solution_analytical) >
     ↪ abs(solution_analytical*0.01)) then
2     newlevel = level + 1
3     [...]
4     if (newlevel > 20) then
5         x(1) = x_end + 1
6         abort1 = .true.
7         return
8     else
9         new_step = step / 2.
10        counter = counter + 1
11        call recursive_step( new_step, newlevel )
```

```
12          call recursive_step( new_step, newlevel )
13      end if
14  [...]
15  end if
```

Listing 3.2: Programmausschnitt aus `DynStep` zum Ablauf der adaptiven (rekursiven) Schrittweitenhalbierung.

Als letztes Programm wurde `DynStep` modifiziert, sodass nach einer häufigen Halbierungsanzahl die zulässige Fehlertoleranz zur analytischen Lösung erhöht wird (siehe Programm `DynStepAcc`). Sobald ein Ausgangsschritt durchlaufen ist, also die integrierte Länge der Schrittweiten `step` gleich der Länge des Ausgangszeitschritts `delta_x` ist, wird die Genauigkeit wieder auf ihren Ausgangswert zurückgesetzt. Das Outputformat des Programms ist das Gleiche wie bei `DynStep`.

3.3 Programmablauf des NPD-Modells

Um die Auswirkungen des Heun- und RK4-Verfahrens in einer praxisorientierten Modellanwendung zu untersuchen, wurde das Programm `npd_mod` von Johannes Pätsch verwendet und angepasst. Das Programm simuliert ein NPD-Modell, welches in zwei Boxen unterteilt ist, die unterschiedliche Schichten in einer Wassersäule darstellen (siehe Abb. 3.2). In beiden Boxen gibt es jeweils eine Variable für das Phytoplankton `st_p` , den Detritus `st_d` und den Nährstoff `st_n` . Die Variablen interagieren dabei durch verschiedene Prozesse: 1) die Aufnahme des Nährstoffs durch das Phytoplankton unter Verwendung der Aufnahmerate `up` , 2) das Bilden von Detritus durch das Absterben des Phytoplanktons unter Verwendung der Mortalitätsrate `di` und 3) die Freisetzung des Nährstoffs durch die Remineralisierung des Detritus unter der Verwendung der Remineralisierungsrate `re` . Daneben gibt es Austauschprozesse zwischen den beiden Boxen: 1) die Diffusion aller Variablen unter Verwendung der Diffusionsrate `diff` und 2) das Absinken des Detritus unter Verwendung der Sinkgeschwindigkeit `w` . Die Implementierung dieser Prozesse ist in Listing 3.3 dargestellt.

Die Aufnahme der Nährstoffe (Photosynthese) findet nur in der oberen Box statt, da diese die euphotische (= lichtdurchflutete) Zone repräsentiert. Die untere Box hingegen stellt die tieferen Schichten dar, in denen kein Licht zur Verfügung steht, welches jedoch zwingend für die Photosynthese benötigt wird. Die Diffusion bewirkt einen Austausch aller Variablen zwischen den beiden Boxen in Richtung des negativen Konzentrationsgradienten (Fick'sches Gesetz) [Wik16c]. Außerdem bewirkt das Sinken einen gerichteten Transport des Detritus aus der oberen Box in die untere.

Die Variablen `st_x(n)` sind die Werte für den vorherigen Zeitschritt der oberen Box (n=1) oder der unteren Box (n=2). Mithilfe der berechneten Variablen `sst_x(n)` , die die Ableitung $f(t_n, y_n)$ an einer Stelle t beschreiben, kann anschließend ein Zwischenschritt bzw. der nächste Wert mit dem jeweiligen Verfahren berechnet werden.

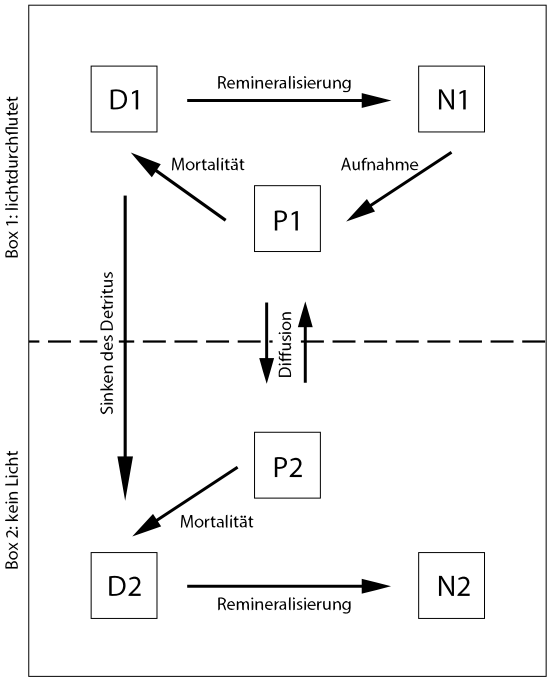


Abbildung 3.2: Schematischer Aufbau und Interaktionen des NPD-Modells.

```
1 sst_p(1) = up*st_p(1)*st_n(1) -di*st_p(1)
    ↪ +diff*(st_p(2)-st_p(1))
2 sst_d(1) = di*st_p(1) -re*st_d(1) -w*st_d(1)
    ↪ +diff*(st_d(2)-st_d(1))
3 sst_n(1) = -up*st_p(1)*st_n(1) +re*st_d(1)
    ↪ +diff*(st_n(2)-st_n(1))
4
5 sst_p(2) =-di*st_p(2) +diff*(st_p(1)-st_p(2))
6 sst_d(2) = di*st_p(2) -re*st_d(2) +w*st_d(1)
    ↪ +diff*(st_d(1)-st_d(2))
7 sst_n(2) = re*st_d(2) +diff*(st_n(1)-st_n(2))
```

Listing 3.3: Programmabschnitt aus `npd_mod` für die Berechnung der verschiedenen Zustandsvariablen in den 2 Modellboxen.

In dem Modell aus Listing 3.3 stellte sich sehr schnell ein Gleichgewicht der Prozesse ein, wodurch sich die drei Variablen nicht mehr änderten. Da Phytoplankton nur unter Vorhandensein von ausreichend Sonnenlicht Photosynthese betreiben kann, wurde der Einfluss des Tag-Nacht-Zyklus sowie des Jahresgangs der Sonneneinstrahlung als zusätzlicher Faktor in den Aufnahmeterm des Phytoplanktons in Box 1 implementiert (`qs`). Dabei wird zu Beginn des Programms das Maximum der Sonneneinstrahlung eines Jahres bestimmt und anschließend zur Normierung des Lichts verwendet. Der zusätzliche

Einfluss unterschiedlicher Wolkenbedeckung wurde dabei vernachlässigt. Damit änderte sich die Berechnung des Phytoplanktons und des Nährstoffs in der oberen Box wie folgt:

```
1 sst_p(1) = up*st_p(1)*st_n(1)*qs -di*st_p(1)
    ↪ +diff*(st_p(2)-st_p(1))
2 sst_n(1) = -up*st_p(1)*st_n(1)*qs +re*st_d(1)
    ↪ +diff*(st_n(2)-st_n(1))
```

Listing 3.4: Berechnung der Zustandsvariable des Phytoplanktons und des Nährstoffs in der oberen Modellbox nach dem Hinzufügen des Einflussfaktors Sonnenlicht.

Zuerst wurde eine Gleichung der *Denning Research Group, Colorado State University* verwendet [Den16], die das Sonnenlicht abhängig vom Breitengrad berechnet. Die berechneten Werte bei 55° nördlicher Breite wichen jedoch signifikant von Vergleichswerten des Earth System Research Laboratory (U.S. Department of Commerce) [Ear16] und der Webseite `www.pveducation.org` [HB16] ab. So lag zum Beispiel das Minimum der implementierten Variante bei ca. 0,06 kW/m^2 und ca. 0,4 kW/m^2 bei den Referenzwerten. Die Maxima liegen bei ca. 0,81 kW/m^2 (*Denning Research Group*) und ca. 1,0 kW/m^2 ([HB16, Ear16]). Da die Abweichungen damit erheblich sind, wurde auf eine Berechnung der Sonneneinstrahlung (siehe Anhang B) aus dem hydrodynamischen Modell HAMSOM (HAMburg Shelf Ocean Model; [Bac85, Poh91, Poh96]) der Universität Hamburg, Institut für Meereskunde zurückgegriffen. Diese Implementation berechnet ebenfalls Werte für einen wolkenlosen Himmel, lieferte allerdings mit einem Minimum von ca. 0,22 kW/m^2 und ein Maximum von ca. 0,9 kW/m^2 erheblich bessere Resultate. Unter diesem Einfluss werden neue Zustandsvariablen berechnet und anschließend entweder alle Lösungen oder für jeden Tag ein Mittelwert in einer Datei abgespeichert.

In einem zweiten Programm `npd_optimal` werden die optimalen Schrittweiten für die Verfahren berechnet. Hierfür wurden zu Anfang zwei Abfragen genutzt. Zum einen wurde abgefragt, ob die neu berechneten Werte größer als ein Epsilon (1^{-30}) sind, zum anderen wurde überprüft, ob die Abweichung des alten zum neuen Punkt kleiner als eine bestimmte relative Abweichung ist. Die erste Abfrage baut darauf auf, dass es keine negativen Bestände an Phytoplankton, Detritus oder Nährstoffen geben kann und somit eine realitätsbezogene Schranke existiert. Die zweite Abfrage wurde aus dem ECOHAM-Programm übernommen, erwies sich aber als nicht sinnvoll wie in Kapitel 4.3.2 gezeigt wird.

Ein drittes Programm `npd_time` misst die Laufzeit der Verfahren und gibt diese auf der Konsole aus. Eine Ausgabe der berechneten Punkte findet nicht statt, da das Abspeichern der Punkte die Laufzeit beeinflussen würde und die Abspeicherung bereits mit dem `npd_mod`-Programm möglich ist. Das Programm beinhaltet nur die Lösungsverfahren und die benötigten Subroutinen.

Die Auswahl des Simulationszeitraums (Anzahl zu berechnender Tage), der Schrittweite und der Prozessparameter sowie die Festlegung der Startwerte für die Konzentrationen von Phytoplankton, Nährstoff und Detritus erfolgt über drei Setup-Dateien, welche beliebig angepasst werden können.

3.4 Korrektheit

Die Korrektheit des `SolverComparison` Programms wurde im Vergleich mit Ergebnissen der semantischen Suchmaschine Wolfram Alpha [Wol16] und exakt berechneten Ergebnissen überprüft. Unter Verwendung einer ausreichend kleinen Schrittweite stimmten die Resultate der Verfahren mit den exakten Lösungen überein. Bei hohen Schrittweiten entsprachen sie denen von Wolfram Alpha berechneten.

Da das Programm `OptimalResults` nur kleinere Anpassungen gegenüber `SolverComparison` beinhaltet, die allerdings keine Auswirkungen auf die berechneten Punkte haben, war keine explizite Korrektheitsüberprüfung notwendig. Unter Verwendung derselben Schrittweite in `SolverComparison` und `OptimalResults` wurden die gleichen Ergebnisse erzielt.

Mithilfe des `OptimalResults` Programms konnte die Korrektheit von `OptimalStep` analysiert werden. Unter Angabe der berechneten optimalen Schrittweiten waren alle berechneten Punkte innerhalb einer einprozentigen Toleranz und in angefertigten Plots untereinander nahezu deckungsgleich.

Zur Korrektheitsüberprüfung von `DynStep` wurde zuerst ohne konstanten Speicherplatzverbrauch gearbeitet und alle berechneten Werte in eine Datei geschrieben. Erneut folgte die Überprüfung im Vergleich mit den exakt berechneten Werten. Mithilfe dieser Ergebnisse konnte anschließend eine Implementation mit Nutzung konstanten Speicherplatzes verifiziert werden. Eine Analyse des `DynStepAcc` Programms wurde auf gleiche Weise durchgeführt.

Da das NPD-Modell eine analytische nicht zu lösende DGL abbildet, wurden die Ergebnisse der Verfahren nach Heun und RK4 mit dem Euler-Verfahren verglichen. Für eine erste Verifikation wurden die Werte ohne Beeinflussung durch das Sonnenlicht gegenübergestellt und mehrere Startparameter getestet. Unter Verwendung kleiner Schrittweiten waren die Ergebnisse der drei Verfahren nahezu deckungsgleich, bei größeren waren zu dem jeweiligen Verfahren typische Abweichungen erkennbar, die durch die verschiedenen Konvergenzordnungen der Verfahren entstehen.

Die Überprüfung des `npd_optimal`-Programms erfolgte wie beim `OptimalStep`. Die berechneten optimalen Schrittweiten wurden in `npd_mod` zur Berechnung der approximierten Funktionswerte verwendet und anschließend aufgetragen. Keine der Variablen wurde dabei negativ.

Eine Verifikation der Zeitmessung `npd_time` war nicht notwendig, da es sich bei dieser um eine reduzierte Variante von `npd_mod` handelt.

Zusammenfassung

Die geschriebenen Programme basieren alle auf der Angabe eines Startpunktes und der Anzahl an Iterationen bzw. eines Endpunktes. Die Bestimmung der optimalen Schrittweite unter Anwendung definierter Kriterien erlaubt es die verschiedenen numerischen Verfahren zur Lösung von DGL hinsichtlich ihrer Genauigkeit zu vergleichen. Dies wurde im Falle

der Analyseprogramme (siehe Kapitel 3.2) für die Verfahren nach Euler, Heun, RK4 und AB2 durchgeführt. Für das NPD-Modell wurden dieselben Verfahren mit Ausnahme von AB2 diesbezüglich analysiert. Bei den Kriterien handelt es sich um eine prozentuale Abweichung (`OptimalStep`) oder bei den `npd`-Programmen um die Abfrage, ob die berechneten Werte größer einem Epsilon sind. Unter Verwendung der `SolverComparison`-, `OptimalResults`- und `npd`-Programme können Genauigkeitsvergleiche angestellt werden. Eine dynamische Zeitschritthalbierung kann mithilfe von `DynStep` und `DynStepAcc` durchgeführt werden. Mit diesen wird getestet, ob adaptive Schritthalbierungen einen Vorteil in bestimmten Berechnungen bringen. Zeitmessungen werden dabei ebenfalls durchgeführt und können im Falle des NPD-Modells mithilfe von `npd_time` durchgeführt werden.

4 Durchführung und Auswertung

In diesem Kapitel wird die Durchführung der Genauigkeits- und Leistungsanalysen beschrieben und die Ergebnisse dieser Analysen vorgestellt. Anschließend werden die Ergebnisse des NPD-Modells unter Verwendung der Verfahren nach Euler, Heun und RK4 verglichen. Zuletzt folgt die Bewertung der erzielten Resultate.

4.1 Testumgebung

Alle Programme wurden auf einem Cluster der Universität Hamburg, Fachbereich Informatik Arbeitsgruppe Wissenschaftliche Rechnen kompiliert und ausgeführt. Das Cluster verfügt über zehn Knoten mit jeweils zwei *Intel Xeon X5650* Prozessoren (jeweils 6 Kerne à 2,66 GHz bzw. 3,06 GHz im TurboBoost mit 2 echten Threads/Kern) und 12 GB Arbeitsspeicher. Als Betriebssystem wird ein 64-bit *Ubuntu 14.04.4 LTS* mit dem Kernel *3.13.0-92-generic x86_64* verwendet. Als Compiler lag gfortran in der Version *4:4.8.2-1ubuntu6* vor.

4.2 Leistungsanalyse

Im Folgenden werden die Zeit- und Genauigkeitsmessungen der geschriebenen Analyseprogramme unter Verwendung zweier Differentialgleichungen, unterschiedlicher Schrittweiten und verschieden großer Intervalle oder Schrittanzahlen vorgestellt und die Ergebnisse analysiert.

4.2.1 Durchführung

Die Programme zur Analyse der Verfahren wurden auf zwei verschiedene DGL angewendet:

$$y'(x) = \frac{1}{2 \cdot y(x)} \tag{4.1}$$

$$y'(x) = sin(x) \cdot y(x) \tag{4.2}$$

Im Folgenden wird Gleichung (4.1) *root* und Gleichung (4.2) *eCos* genannt. Anfangswerte waren jeweils die zur Laufzeit mit doppelter Genauigkeit berechneten exakten y-Werte an $x = 1$.

Die ersten Messungen mit dem `SolverComparison`-Programm erfolgten mit unterschiedlich großen Schrittweiten, damit die Genauigkeit der verschiedenen Verfahren dargestellt werden konnte. Anschließend wurden mittels berechneter optimaler Schrittweiten und dem `OptimalResults`-Programm Laufzeitmessungen durchgeführt.

Die Analyse der dynamischen Schrittweitenanpassung erfolgte zum einen für mehrere Schrittweiten, zum anderen mittels einer stetigen Intervallvergrößerung. Alle Programme mit Ausnahme des `OptimalStep`-Programms sind in jeweils drei Unterordner aufgeteilt. Es existiert jeweils ein Ordner mit dem Fortran-Code `fortran`, ein Ordner `matlab`, der die Skripte zum Plotten enthält und ein `Output`-Ordner, der Unterordner namens `ascii` und `figures` enthält. Nach Ausführung der kompilierten Fortran-Datei werden die Ergebnisse als csv-Datei im Ordner `ascii` hinterlegt. Mithilfe des jeweiligen MATLAB-Skriptes werden aus diesen ASCII-Daten Abbildungen erzeugt, welche im Ordner `figures` gespeichert werden.

Tabelle 4.1 liefert eine Übersicht über die verschiedenen Einstellungen, die für die Analyse in Kapitel 4.2.2 verwendet wurden. Der Startwert `x_start` war bei allen Analysen gleich eins und die Iterationen geben die Anzahl der Erhöhung von `delta_x` und `x_end` während der Ausführung des jeweiligen Programms an. Die Kürzel in der ersten Spalte werden im Folgenden für die Beschreibung des jeweiligen Satzes an Einstellungen (Spalten 2-7) verwendet.

Tabelle 4.1: Kürzel für gewählte Einstellungen der Laufzeit- und Genauigkeitsvergleiche.

Kürzel	Funktion	`delta_x`	`nLocs_x`	`x_end`	Iterationen	erlaubte Abw.
Func1	*root*	20	10	-	1	-
Func2	*eCos*	0,05	2000	-	1	-
Optimal	*root*	siehe Tab. 4.2	-	5^6 – 5^7	10	1 %
Delta1	*root*	1 – 381	-	10^8	20	1 %
Delta2	*eCos*	5^{-4} – $2,4^{-3}$	-	100	20	2 %
DeltaAcc	*eCos*	7^{-3}	-	100	1	1 %

Tabelle 4.2: Optimale Schrittweiten ermittelt mit `OptimalStep` (*root*-DGL).

Verfahren	Optimale Schrittweite
Euler	0,3125
RK4	5
AB2	0,625
Heun	2,5

4.2.2 Laufzeit- und Genauigkeitsvergleich

Im Folgenden wird nur auf einige erstellte Grafiken eingegangen, die besonders anschaulich die Ergebnisse darstellen. Weitere Grafiken und zugehörige Wertetabellen finden sich auf der beiliegenden CD.

Zunächst sollen die Unterschiede in der Genauigkeit und Leistung der Verfahren unter Verwendung der *root*-DGL gezeigt werden. Dafür wird in Abb. 4.1 der Vergleich zwischen der analytischen Lösung (blau) und den numerisch approximierten Lösungen nach den Verfahren Euler (grün), RK4 (rot), AB2 (orange) und Heun (violett) für *Func1* abgebildet. Die Farbzuordnungen bleiben in den folgenden Abbildungen gleich.

In der Abbildung fallen besonders das Euler-Verfahren durch seine hohe Abweichung und der erste Schritt des AB2-Verfahrens auf. Während der erste Schritt des AB2-Verfahrens mit dem des Euler-Verfahrens übereinstimmt, liefert es ab dem zweiten Schritt deutlich genauere Approximationen. Unter genauer Betrachtung ist ebenfalls zu erkennen, dass das AB2-Verfahren eine kleine Verbesserung gegenüber dem Heun-Verfahren darstellt. Am besten approximiert allerdings das RK4-Verfahren die DGL, was insbesondere am ersten Schritt sehr gut sichtbar wird.

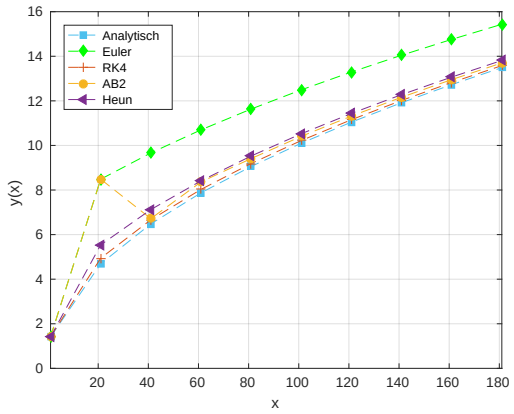


Abbildung 4.1: *Func1*, Genauigkeitsvergleich der numerisch approximierten zur analytischen Lösung. Erster Schritt des Euler- und AB2-Verfahrens ist deckungsgleich.

Zur Analyse des Zusammenhangs von Genauigkeit und gewählter Schrittweite wurde eine Laufzeitanalyse mit den *Optimal*-Werten (siehe Abb. 4.2) durchgeführt. Dabei wird die Laufzeit gegen die Intervallbreite aufgetragen.

Der Laufzeitunterschied zwischen den Paaren RK4 und Heun zu Euler und AB2 ist offensichtlich. Hierbei sind RK4 und Heun etwa drei bis viermal schneller als Euler und AB2. Untereinander ist die Laufzeit des AB2-Verfahrens besser als die des Euler-Verfahrens, da die Schrittweite zweimal so groß ist und ein ähnlicher Aufwand in der Berechnung vorliegt. Das RK4-Verfahren ist sehr genau und benötigt daher nur die doppelte Schrittweite von Heun. Da es allerdings zwei zusätzliche Zwischenschritte pro Berechnungsschritt berechnet, sind beide Verfahren in den getesteten Intervallen etwa gleich schnell.

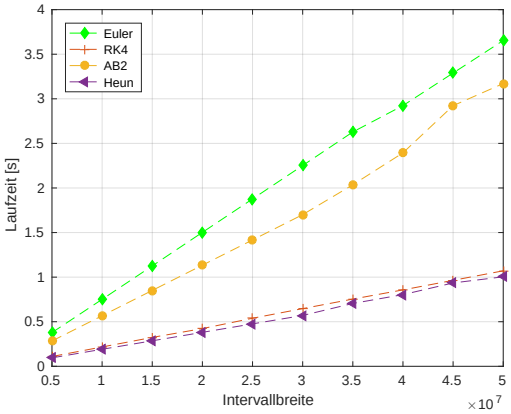


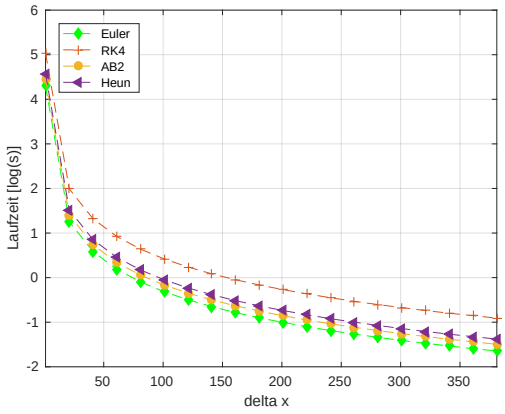
Abbildung 4.2: *Optimal*, Laufzeitanalyse numerischer Lösungsverfahren bei stetig ansteigender Intervallgröße.

Für die Untersuchung der Verfahren unter Verwendung der adaptiven Schrittweitenhalbierung wurden die Werte *Delta1* genutzt (siehe Tabelle 4.1). Abbildung 4.3 gibt eine Übersicht über Laufzeit (Abb. 4.3a), Anzahl benötigter Halbierungen (Abb. 4.3b) und die maximale Abweichung zur analytischen Lösung (Abb. 4.3c) in Abhängigkeit von der Schrittweite.

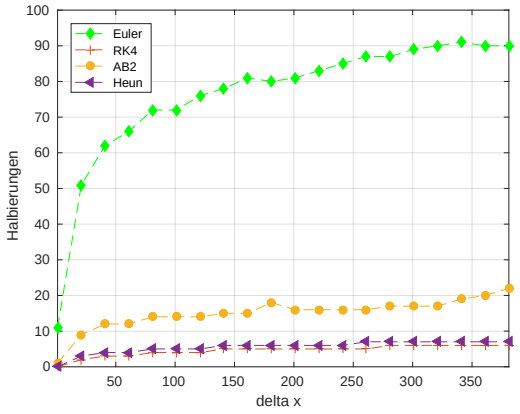
Aus den Abbildungen lässt sich ablesen, dass trotz einer höheren Anzahl an Halbierungen – die durch größere Fehler entsteht (Euler und AB2 sind deckungsgleich) – das Euler-Verfahren die Lösung der *root*-DGL am schnellsten approximiert. Jedoch ist die Funktion unter hoch gewählter Schrittweite mit jedem Verfahren sehr schnell zu berechnen, da die Anzahl der Halbierungen nahezu konstant bleibt. Dies hat die Ursache, dass die Lösung der DGL nach wenigen Iterationen nahezu linear verläuft und damit sehr einfach zu approximieren ist. Durch Berechnung der zusätzlichen Zwischenschritte des Heun- und RK4-Verfahrens sind die Lösungen zwar genauer, bei einer nahezu gleich bleibenden Steigung ist dies aber nicht notwendig. Die Deckungsgleichheit des Euler- und AB2-Verfahren lässt darauf schließen, dass die maximale Abweichung im ersten Schritt auftritt, da dieser bei beiden Verfahren identisch ist. Dennoch benötigt das AB2-Verfahren sechs bis siebenmal weniger Halbierungen durch die höhere Genauigkeit. Einen Laufzeitvorteil bietet AB2 allerdings gegenüber Euler nicht, da das Verhältnis von Anzahl der Halbierungen zu höherem Berechnungsaufwand nicht ausreichend ist. Auch das Heun- und RK4-Verfahren zeigen im Falle der *root*-DGL keine Laufzeitverbesserung gegenüber dem Euler-Verfahren und weisen eine höhere Laufzeit als dieses auf. Je höher dabei die Ordnung des Lösungsverfahrens ist, desto langsamer wird die Lösung approximiert.

Die Anzahl der Halbierungen aller Verfahren steigt in Abb. 4.5b nur langsam. Ursache dafür ist, dass für mehrere Schrittweiten und damit unterschiedliche maximale Fehler, die gleiche Halbierungsanzahl benötigt werden kann. Eine Halbierung deckt also eine größere Fehlerspanne ab. Die Halbierungsanzahl erhöht sich nur dann signifikant, wenn die DGL trotz der Halbierungen nicht mehr genau genug approximiert werden kann. Wird die maximale Abweichung zur analytischen Lösung verringert, müssen das Euler- und das AB2-Verfahren in den ersten Iterationen der Berechnung häufiger ihre Schritte

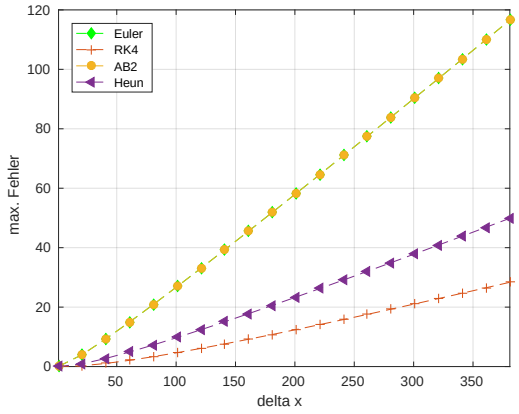
halbieren, da der Fehler zu groß ist. Da die Lösung der *root*-DGL nach dem ersten Anstieg allerdings nahezu linear verläuft, sind kaum weitere Schritthalbierungen nötig. Ist das zu berechnende Intervall relativ zur Schrittweite gesehen klein, bietet ein Verfahren mit höherer Ordnung als das Euler-Verfahren einen Vorteil gegenüber diesem. Ist das Intervall hingegen breit (ca. 10.000 mal größer als die Schrittweite), ist die Approximation durch das Euler-Verfahren schneller.



(a) Laufzeit



(b) Anzahl an Halbierungen



(c) Maximale Abweichung zur analytischen Lösung - Euler und AB2 deckungsgleich

Abbildung 4.3: *Delta1*: Laufzeit, Anzahl benötigter Halbierungen und maximale Abweichung zur analytischen Lösung in Abhängigkeit von der Schrittweite `delta_x`.

Zur Analyse der Leistung und Genauigkeit der Approximation einer komplexeren DGL folgen Auftragungen unter Nutzung der *eCos*-DGL. Zuerst wird hierbei in Abbildung 4.4 die analytische Lösung gegen den vier Approximationen mit den Einstellungen *Func2* aufgetragen (siehe Tabelle 4.1). Da 2000 Punkte berechnet wurden, werden diese der

Übersichtlichkeit halber nicht mit dargestellt. Die AB2-, Heun- und RK4-Verfahren sind deckungsgleich.

Wiederum ist die deutliche Abweichung der Approximation durch Euler auffallend, welche mit jeder Schwingung zunimmt. Auch kann man erkennen, dass keines der Verfahren die Maxima genau approximieren kann. Dies liegt bei Heun und RK4 vermutlich daran, dass die berechneten Zwischenschritte bereits die negative Steigung mit in die Berechnung des nächsten Punktes einbeziehen. Beim AB2-Verfahren kann die Abweichung im Grunde nur durch eine geringere Genauigkeit gegenüber dem Heun- und RK4-Verfahren entstanden sein, da bei diesem für die Bestimmung des nächsten Punktes kein zusätzlicher nächster Punkt verwendet wird.

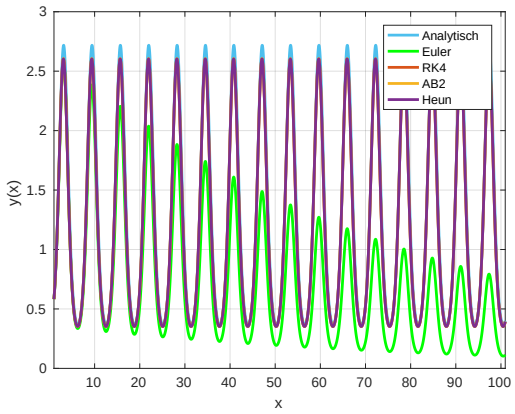


Abbildung 4.4: *Func2*: Genauigkeitsanalyse der numerisch approximierten zur analytischen Lösung - AB2, Heun und RK4 deckungsgleich.

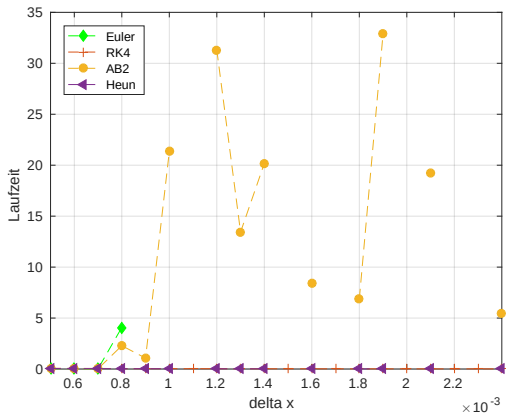
Die optimalen Schrittweiten der Verfahren zur Lösung der *eCos*-DGL sind in Tabelle 4.3 angegeben. Dabei fällt auf, dass die optimalen Schrittweiten für das AB2-, Heun- und RK4-Verfahren jeweils gleich sind. Dies hat vermutlich die Ursache, dass zur Bestimmung der optimalen Schrittweiten eine Standardschrittweite solange halbiert wird, bis die Abweichung in dem erlaubten Rahmen ist. Würden die Schrittweiten nicht halbiert, sondern z. B. ein sehr kleiner Wert von ihnen subtrahiert, würden verschiedene optimale Schrittweiten festgestellt werden. Dies gilt vermutlich ebenfalls für das Euler-Verfahren und ist der Grund für die gleichen optimalen Schrittweiten bei beiden erlaubten Abweichungen. Verwendet wird die Subtraktion der Schrittweite zur Bestimmung der optimalen Schrittweiten allerdings nicht, da sie die Laufzeit übermäßig beeinflusst.

Tabelle 4.3: Optimale Schrittweiten ermittelt mit `OptimalStep` (*eCos*-DGL).

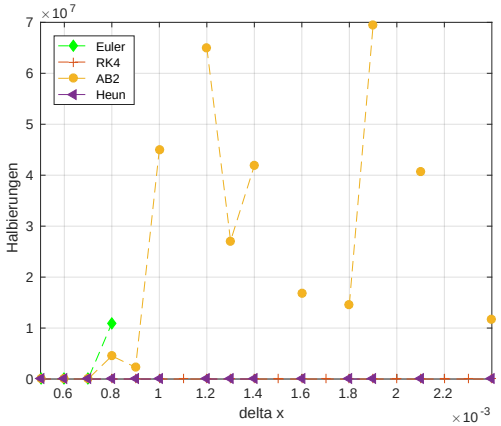
Verfahren	Opt. Schrittweite (1 % erlaubte Abw.)	Opt. Schrittweite (2 % erlaubte Abw.)
Euler	6.103516E-04	6.103516E-04
RK4	4.882813E-03	9.765625E-03
AB2	4.882813E-03	9.765625E-03
Heun	4.882813E-03	9.765625E-03

Ebenfalls ist in der Tabelle zu erkennen, dass die Schrittweite des Euler-Verfahrens bei erlaubter einprozentiger Abweichung achtmal und bei der zweiprozentigen 16 mal kleiner als die der anderen Verfahren ist. Dies zeigt insbesondere die Ungenauigkeit des Euler-Verfahrens bei komplexeren DGLs wie der *eCos*-DGL.

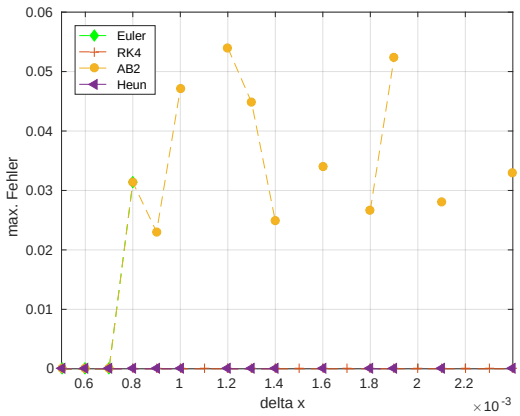
In der vorhergehenden Abb. 4.4 wurde eine 2 bis 1,5 mal höhere Schrittweite als bei Abb. 4.5 verwendet, da durch diese die deutliche Abweichung des Euler-Verfahrens sichtbar wird. Bei kleineren Schrittweiten ist die Abweichung noch immer vorhanden, lässt sich allerdings schlechter zeigen.



(a) Laufzeit



(b) Anzahl an Halbierungen



(c) Maximale Abweichung zur analytischen Lösung - Euler und AB2 bei den ersten vier Schrittweiten deckungsgleich

Abbildung 4.5: *Delta2*: Laufzeit, Anzahl benötigter Halbierungen und maximale Abweichung zur analytischen Lösung in Abhängigkeit von der Schrittweite `delta_x` - alle Verfahren mit den ersten drei Schrittweiten deckungsgleich.

In Abbildung 4.5c lässt sich gut erkennen, dass nur das RK4-Verfahren eine fehlerfreie Approximation der Lösung ermöglicht. Am zweitbesten schneidet das Heun-Verfahren ab und kann bei 14 von 20 Schrittweiten ohne Abweichung zur analytischen Lösung

approximieren. In den anderen Fällen ist keine Approximation innerhalb der zugelassenen Abweichung möglich. Das AB2-Verfahren schafft die Berechnungen mit den gleichen Schrittweiten wie das Heun-Verfahren, weicht allerdings von der Lösung ab. Die Berechnung durch Euler ist nur mit den ersten vier Schrittweiten möglich und hat bei diesen den gleichen Fehler wie das AB2-Verfahren. Während Euler bei diesem Fehler über 10^7 Halbierungen benötigt, sind es bei AB2 nur etwa die Hälfte. Hierdurch zeigt sich besonders der Genauigkeitsvorteil des AB2-Verfahrens. Für die höheren Schrittweiten benötigt AB2 bis zu $7 \cdot 10^7$ Halbierungen. Dies erfordert sichtbar mehr Zeit im Vergleich zu den genaueren Heun- und RK4-Verfahren, die bei den gleichen Schrittweiten keine Halbierungen benötigen (Abb. 4.5a). Durch die Messergebnisse stellte RK4 sich als das schnellste der untersuchten Verfahren heraus, da die Schrittweite unter gleichbleibender Genauigkeit deutlich erhöht werden konnte, wodurch weniger Punkte berechnet werden mussten.

Unter Verwendung des `DynStepAcc`-Programms wurden die Verfahren mit den Einstellungen *DeltaAcc* ausgeführt (siehe Tabelle 4.4). Dabei wurde nach zwanzig Halbierungen die zulässige Abweichung erhöht. Das Heun-Verfahren stellte sich als das schnellste heraus und brauchte etwas weniger Zeit als das RK4-Verfahren. Das Euler-Verfahren war das drittschnellste und das AB2-Verfahren mit Abstand das langsamste. Die Laufzeiten lassen sich auf die Anzahl der Halbierungen und Abweichungserhöhungen in der dritten bzw. sechsten Spalte zurückführen. Während Heun und RK4 eine sehr geringe Anzahl an Erhöhungen aufweisen, also auch weniger Schritthalbierungen durchführen müssen, brauchen AB2 und insbesondere Euler deutlich mehr Erhöhungen. Euler hat dabei auch die von allen Verfahren höchste maximale Abweichung, wodurch die Lösung nur ungenau dargestellt werden kann. AB2 hingegen benötigt aufgrund der außerordentlich hohen Anzahl an Halbierungen zwar etwa sechsmal so viel Zeit wie Euler, ist dabei aber um bis zu 5 % genauer. Die hohe Anzahl an Halbierungen entsteht dadurch, dass das AB2-Verfahren in nahezu jedem Schritt viele Halbierungen benötigt, deshalb aber eine genauere Approximation als das Euler-Verfahren liefert und damit weniger Erhöhungen der Abweichung erforderlich sind.

Tabelle 4.4: *DeltaAcc*, Leistungsanalyse der Lösungsverfahren mit Erhöhung der Abweichung zur analytischen Lösung bei häufigen Halbierungen (n>20) eines Schritts.

Verfahren	Laufzeit [s]	Halbierungen	max. Fehler	max. Abw.	Erhöhungen d. Abw.
Euler	38,39	39.778.269	0,163796	0,07	42.651
RK4	0,15	70.952	0,022422	0,02	3.709
AB2	233,16	186.566.414	0,039981	0,02	9.031
Heun	0,12	69.832	0,022460	0,02	3.653

Ursache für die bessere Laufzeit von Heun gegenüber RK4 sind die geringere Anzahl an Erhöhungen der Abweichung zur analytischen Lösung und die geringere Anzahl der Halbierungen. Diese haben vermutlich wieder ihre Ursache in den Extrempunkten der Lösung. Es lässt sich damit zeigen, dass die Größenordnung der Abweichung eine

bedeutende Rolle in der erreichten Laufzeit spielt und das ein Verfahren mit kleinerer Ordnung mindestens genauso schnell in der Berechnung sein kann, wie eines mit höherer Ordnung. Zu sehen ist dies an dem ähnlich ausfallenden Fehler und der Halbierungsanzahl des Heun- und RK4-Verfahrens.

4.3 NPD-Modell

Im Weiteren werden die Ergebnisse der Analysen des NPD-Modells vorgestellt. Diese wurden unter Verwendung von verschiedenen Startwerten für Phytoplankton, Detritus und Nährstoff durchgeführt.

4.3.1 Durchführung

Für die Vergleiche vom Euler- zum Heun- und RK4-Verfahren wurden die Startwerte für Phytoplankton, Detritus und Nährstoff wie folgt gewählt: Zu Beginn des Jahres ist Winter, deshalb gibt es noch kleine Bestände lebenden Phytoplanktons (P=2). Der Großteil des Phytoplanktons ist allerdings bereits abgestorben (D=5) und davon sind wiederum Anteile remineralisiert (N=5). Mit dem `npd_optimal`-Programm wurden anschließend die optimalen Schrittweiten bestimmt (siehe Tabelle 4.5). Mithilfe des `npd`-Programms und den optimalen Schrittweiten erfolgte dann die Berechnung der Werte für ein theoretisches Jahr (1. Januar bis 31. Dezember). Zum Schluss wurden die Werte mit MATLAB aufgetragen und die Laufzeit mithilfe von `npd_time` gemessen. Die Parameter wie Aufnahme, Mortalität, etc. wurden für die Startwertkonfiguration wie in Tabelle 4.6 gewählt.

Tabelle 4.5: Optimale Schrittweiten ermittelt mit `OptimalStep` (NPD-Modell).

Verfahren	Optimale Schrittweite
Euler	39
RK4	66
Heun	53

Tabelle 4.6: Parameter des NPD-Modells.

Prozess	Parameter [1/s]
Aufnahmerate	0,010
Mortalitätsrate	0,001
Diffusionsrate	0,001
Sinkrate	0,010
Remineralisierungsrate	0,001

4.3.2 Leistungsanalyse

Da das Sonnenlicht den einzigen externen Antrieb des NPD-Modells darstellt, sind in Abbildung 4.6 die Maxima des normierten Sonnenlichts gezeigt. Während im Winter wenig Sonnenlicht zur Verfügung steht, ist das Maximum in der Mitte des Jahres ca. viermal so hoch. Somit stellt dieses einen großen Einfluss auf die Nährstoffaufnahme/Photosynthese des Phytoplanktons in der oberen Box dar.

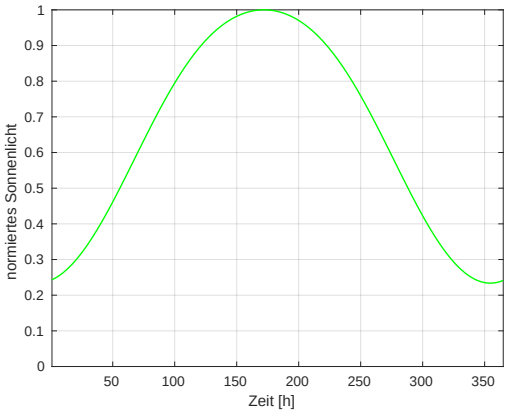
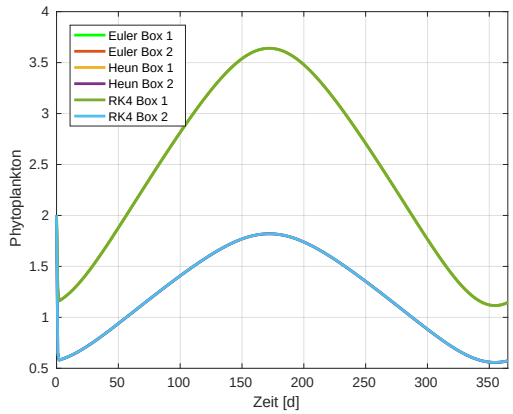


Abbildung 4.6: Maxima der normierten Sonneneinstrahlung über ein Jahr.

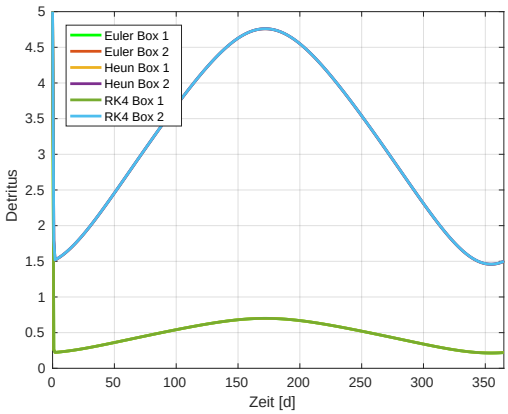
Unter Verwendung der Parameter aus Tabelle 4.6 und Schrittweiten $\Delta t = 1$, wurden für spätere Analysen Vergleichsresultate erstellt. Durch die, im Vergleich zu den optimalen, gering gewählten Schrittweiten stellen diese einen „idealen“ Verlauf der Variablen dar. In Abb. 4.7 werden die gemittelten Konzentrationen für jeden Tag eines Jahres für das Phytoplankton (Abb. 4.7a) den Detritus (Abb. 4.7b) und den Nährstoff (Abb. 4.7c) gezeigt. Dabei sind die Approximationen der drei Verfahren deckungsgleich.

In Abbildung 4.7 fällt auf, dass die gemittelten Konzentrationen des ersten Tages abnehmen (Abbildungen 4.7a und 4.7b) bzw. deutlich ansteigen (Abb. 4.7c). Ursache dafür sind die Anfangsbedingungen, die nicht den Mengenverhältnissen entsprechen, die einem Gleichgewicht bei den vorliegenden Lichtverhältnissen gleich kommen und die hohen Umsatzraten zwischen den einzelnen Variablen. Durch das geringe Sonnenlicht kann das Phytoplankton nur in geringen Mengen Nährstoff aufnehmen, deshalb sinkt die Phytoplanktonkonzentration und in Folge dessen steigt die Nährstoffkonzentration an. Der Detritus nimmt ab, da der Phytoplanktonbestand bereits niedrig und deshalb auch die Mortalität gering ist.

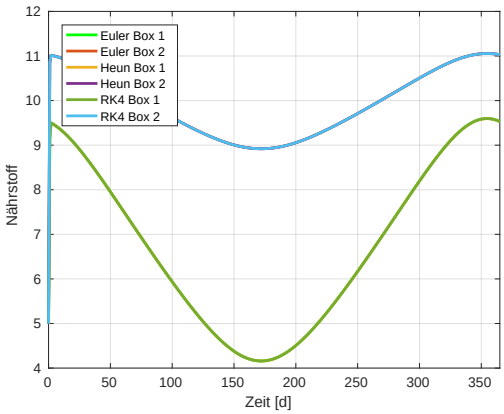
Bis zur Mitte des Jahres steigen bzw. sinken die Konzentrationen der oberen (ersten) Box des Phytoplanktons und Nährstoffs deutlich, da das Sonnenlicht ein rasches Wachstum des Phytoplanktons ermöglicht und deshalb auch mehr Nährstoff verbraucht wird. Der Detritus steigt hingegen in der unteren (zweiten) Box zur Mitte des Jahres an. Grund dafür ist der sinkende Detritus aus der oberen Box und eine hohe Mortalität des Phytoplanktons. Durch die Diffusion zwischen den Boxen findet stetig ein Austausch der Variablen statt, sodass insbesondere das Phytoplankton auch in der unteren Box in großen Mengen vorhanden ist.



(a) Verlauf des Phytoplanktons



(b) Verlauf des Detritus



(c) Verlauf des Nährstoffs

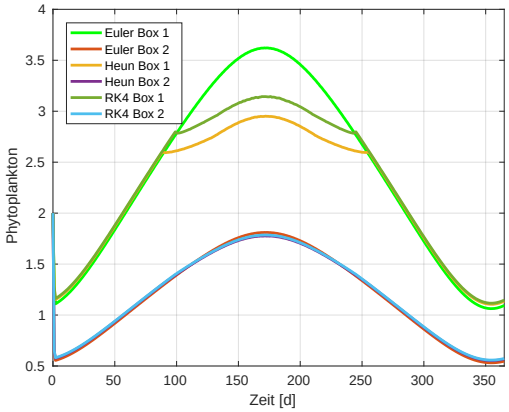
Abbildung 4.7: Verlauf der Variablen des NPD-Modells eines Jahresgangs unter Verwendung der Schrittweite gleich eins für alle Verfahren - Verfahren sind deckungsgleich.

Wie sich durch den Verlauf der Sonneneinstrahlung vermuten lässt, sinken die gemittelten Konzentrationen des Phytoplanktons und des Detritus gegen Ende des Jahres, während der Nährstoff wieder ansteigt. Im Hinblick auf beide Boxen ist beim Phytoplankton und Nährstoff in der oberen Box ein größerer Anstieg und bei dem Detritus in der unteren Box zu sehen. Damit zeigt sich, dass die Sonneneinstrahlung einen großen Einfluss auf das gesamte Modell ausübt, obwohl sie lediglich auf die Nährstoffaufnahme in der oberen Box einen direkten Einfluss hat.

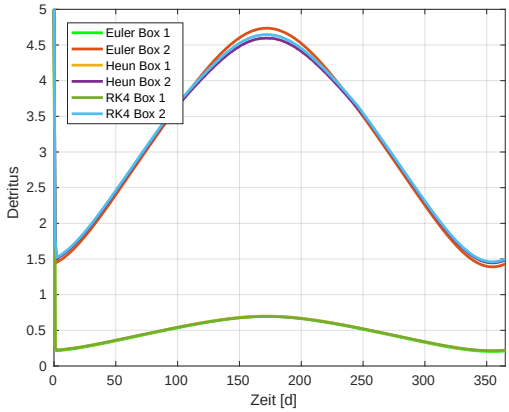
Unter Nutzung der optimalen Schrittweiten (siehe Tabelle 4.5) wurden neue Auftragungen erstellt. Wiederum wird der gemittelte Verlauf des Phytoplanktons (Abb. 4.8a), des Detritus (Abb. 4.8b) und des Nährstoffs (Abb. 4.8c) eines Jahresgangs dargestellt. Dabei wird in hellgrün und rot die Approximationen durch das Euler-Verfahren in der ersten bzw. zweiten Box gekennzeichnet. Das Heun-Verfahren ist durch die Farben orange

(Box 1) und violett (Box 2), und das RK4-Verfahren durch dunkelgrün (Box 1) sowie hellblau (Box 2) markiert.

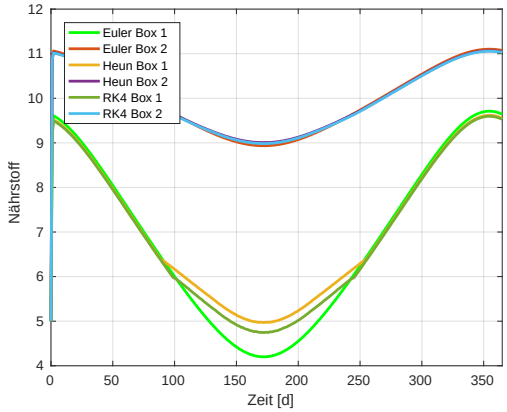
Im Verlauf des Phytoplanktons und Nährstoffs fallen sofort die deutlichen Abweichungen in der oberen Box im Falle des Heun- und RK4-Verfahrens im Vergleich zu der Abb. 4.7 auf. Während das Euler-Verfahren die Lösung sichtbar gut approximieren kann, weicht die Approximation durch das Heun-Verfahren beim Phytoplankton um ca. 30 % und beim Nährstoff um knapp 15 % ab. Die Approximation durch das RK4-Verfahren weicht um ca. 12 % beim Phytoplankton und beim Nährstoff um ca. 10 % ab. Unter genauer Betrachtung des Phytoplanktons (Abb. 4.8a) kann an den Tagen, an denen die deutlichen Abweichungen anfangen, eine Abnahme der Phytoplanktonkonzentration festgestellt werden.



(a) Verlauf des Phytoplanktons



(b) Verlauf des Detritus

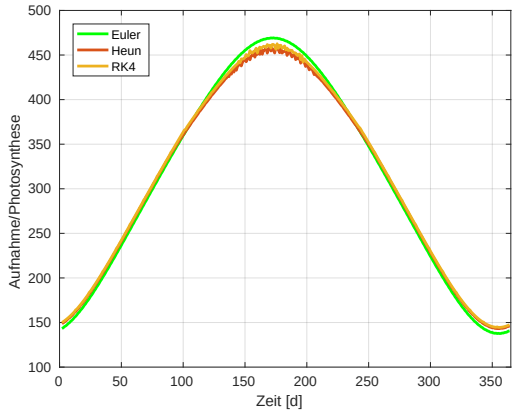


(c) Verlauf des Nährstoffs

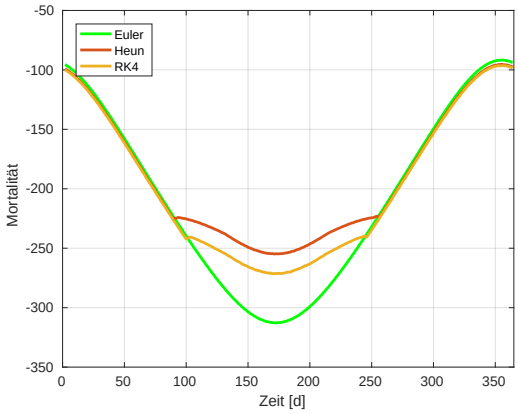
Abbildung 4.8: Verlauf der Variablen des NPD-Modells eines Jahresgangs unter Verwendung der optimalen Schrittweiten - Phytoplankton und Nährstoff in der unteren (zweiten) Box und Detritus in der oberen (ersten) Box deckungsgleich.

Der Verlauf des Detritus der ersten Box wird von allen drei Verfahren deckungsgleich berechnet und stimmt mit den „idealen“ Werten (siehe Abb. 4.7) überein. Auch in der unteren (zweiten) Box wird von allen drei Verfahren der Detritus mit hoher Genauigkeit approximiert. Dabei ist allerdings entgegen zur oberen Box eine geringe Abweichung erkennbar, die beim Heun-Verfahren am höchsten ist. Bei dieser Abweichung handelt es sich um etwa 6 %, die damit um ein Fünftel kleiner als beim Phytoplankton und um ca. die Hälfte kleiner als beim Nährstoff ist. Die Approximationen des Phytoplanktons und des Nährstoffs werden durch alle Verfahren in der unteren Box nahezu deckungsgleich bestimmt. Zum Anfang und Ende des Jahres sind die gemittelten Werte durch das Heun- und RK4-Verfahren etwas höher als die des Euler-Verfahrens. Dies lässt sich ebenfalls in beiden Boxen der Abbildung 4.8 mit der Ausnahme des Detritus in der oberen Box erkennen.

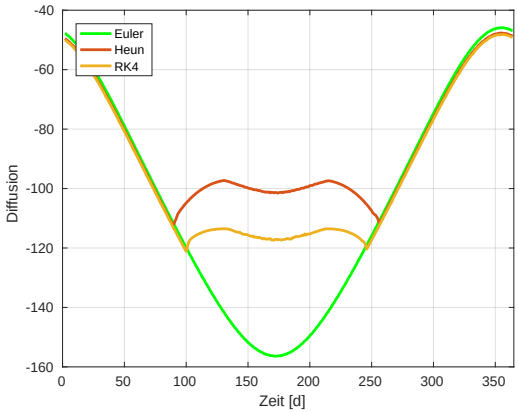
Um die in Abb. 4.8 gezeigten Abweichungen analysieren zu können, werden in Abb. 4.9 die das Phytoplankton beeinflussenden Prozesse dargestellt.



(a) Prozess: Aufnahme/Photosynthese



(b) Prozess: Mortalität

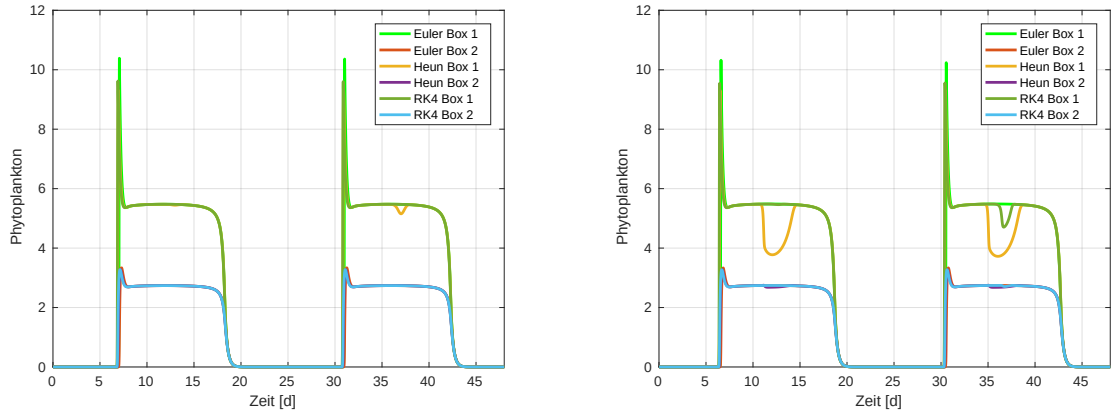


(c) Prozess: Diffusion

Abbildung 4.9: Jahresgang der täglich aufsummierten Prozesse zur Berechnung der Phytoplanktonkonzentration der oberen (ersten) Box.

Jede Änderung eines Prozesses wurde über einen Tag, angefangen bei 12 Uhr Mittags bis 12 Uhr Mittags des nächsten Tages, zeitlich integriert. Hierbei wird zugunsten der Übersichtlichkeit und da die größten Abweichungen in Box 1 auftreten, nur auf die Prozesse in der oberen Box eingegangen. Abgebildet werden damit die Prozesse Aufnahme/Photosynthese (Abb. 4.9a), Mortalität (Abb. 4.9b) und Diffusion (Abb. 4.9c). Die Euler-Approximation ist hellgrün, Heun rot und RK4 orange.

Durch die abgebildete Aufnahme/Photosynthese stellt sich heraus, dass die Approximationen der Verfahren Heun und RK4 beim Maximum nur um 4 % bzw. 3 % von der eulerschen und „idealen“ Lösung abweichen. Die Näherung der Mortalität durch das Heun-Verfahren weicht dagegen um ca. 19 % und durch RK4-Verfahren um 13 % ab. Mit ca. 50 % bzw. ca. 40 % sind die Abweichungen durch das Heun- bzw. RK4-Verfahren bei der Diffusion prozentual am höchsten. Damit wird der Schluss gezogen, dass die Mortalität und die Diffusion die Hauptursachen der hohen Abweichungen sind, wobei die Diffusion sogar rückläufig ist. Da die Abweichungen zur Jahresmitte immer deutlicher werden und im Winter kaum vorhanden sind, spielt die Sonneneinstrahlung vermutlich eine Rolle. Ursprung der geringen Diffusion und Mortalität kann nach der Berechnungsvorschrift (siehe Listing 3.4) allerdings nur eine zu kleine Konzentration des Phytoplanktons der oberen Box selbst sein. Für die genaue Bestimmung der Ursache wurden die Tage ermittelt, an denen die ersten Abweichungen auftraten. Für Heun ist dies an Tag 89 der Fall, für RK4 an Tag 100 (siehe Abb. 4.10).



(a) Tag 88/89: Verlauf des Phytoplanktons - Abweichung des Heun-Verfahrens

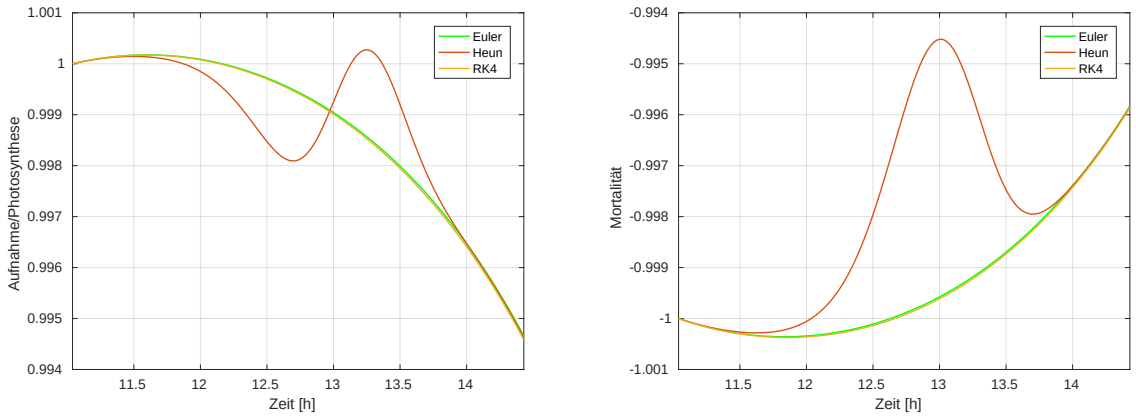
(b) Tag 99/100: Verlauf des Phytoplanktons - Abweichung des RK4-Verfahrens

Abbildung 4.10: Verlauf des Phytoplanktons an den ersten Tagen der auftretenden Abweichung in der Approximation durch das Heun- und RK4-Verfahren

In der Abbildung wird jeweils der Tag mit der ersten deutlichen Abweichung und der vorhergehende gezeigt. Es lässt sich gut erkennen, dass sowohl das Heun- als auch das RK4-Verfahren die Maxima ungenau approximieren. Grund hierfür ist vermutlich die Vorhersage nächster Punkte in Form der Zwischenschritte. Durch diese fließt nicht nur eine Annäherung mit positiver Steigung, sondern außerdem bereits ein nächster Punkt

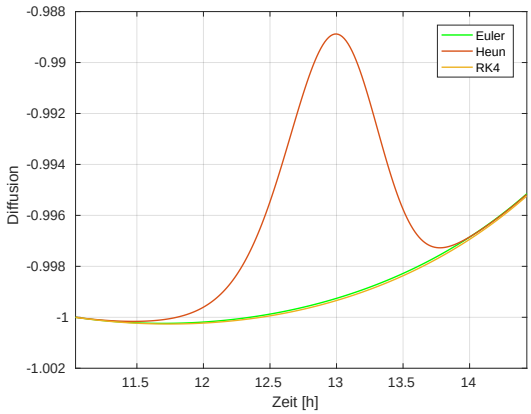
mit negativer Steigung in die Berechnung mit ein. Die Ableitung ändert sich dabei so schnell, dass die Extremstellen nicht richtig durch diese Verfahren angenähert werden können.

Besonders auffallend ist ebenfalls die Abweichung in der Mitte des 89. bzw. 100. Tages. Im Vergleich der beiden Abbildungen 4.10a und 4.10b fällt auch auf, dass die Abweichung zur Mitte des Jahres immer größer wird. Da diese Abweichungen nicht mithilfe der gezeigten Abbildungen erklärbar sind, wurden die Approximationen der Prozesse des Phytoplanktons mit dem Heun-Verfahren im Vergleich zu Euler und RK4 im Zeitintervall des erstmaligen Auftretens des Fehlers (ca. 11:30 bis 14:00 Uhr) aufgetragen (siehe Abb. 4.11).



(a) Prozess: Aufnahme/Photosynthese

(b) Prozess: Mortalität



(c) Prozess: Diffusion

Abbildung 4.11: Tag 88: Normierte Prozesse zur Bestimmung des Phytoplanktons. Fehlerursache der Berechnung mit dem Heun-Verfahren.

Da die Prozesswerte für die drei Verfahren unterschiedlich hoch sind (Nährstoffaufnahme bei Heun bis zu 30 % und bei RK4 bis zu 70 % größer als bei Euler), wurden diese durch den jeweils ersten Funktionswert im gezeigten Intervall normiert. Trotz unterschiedlich hoher Prozesse werden die Konzentrationen der Variablen außer in der Mitte

des Jahres deckungsgleich bestimmt (siehe Abb. 4.8), wodurch die Korrektheit gezeigt wird. Ursache der verschiedenen Prozesswerte sind die berechneten Zwischenschritte des Heun- und RK4-Verfahrens.

Abbildung 4.11c zeigt den Prozess der Diffusion. In dieser ist zeitlich gesehen (zwischen ca. 11:50 und 13:45 Uhr) als Erstes eine Abweichung zur Lösung nach Euler und RK4 erkennbar. Da die Diffusion nur aus der Phytoplanktonkonzentration der oberen und unteren Box berechnet wird, wurden diese überprüft, es konnte aber keine Ursache für die Abweichung festgestellt werden.

Unmittelbar nach der Abweichung der Diffusion, steigt die Mortalität signifikant an und weicht damit ebenfalls von der Approximation des Euler- und RK4-Verfahrens ab. Um etwa 13 Uhr erreichen die Mortalität und die Diffusion ihr Maximum. Anschließend sinken beide merklich ab, um etwa eine halbe Stunde später wieder auf die Werte des Euler- und RK4-Verfahrens abzusinken.

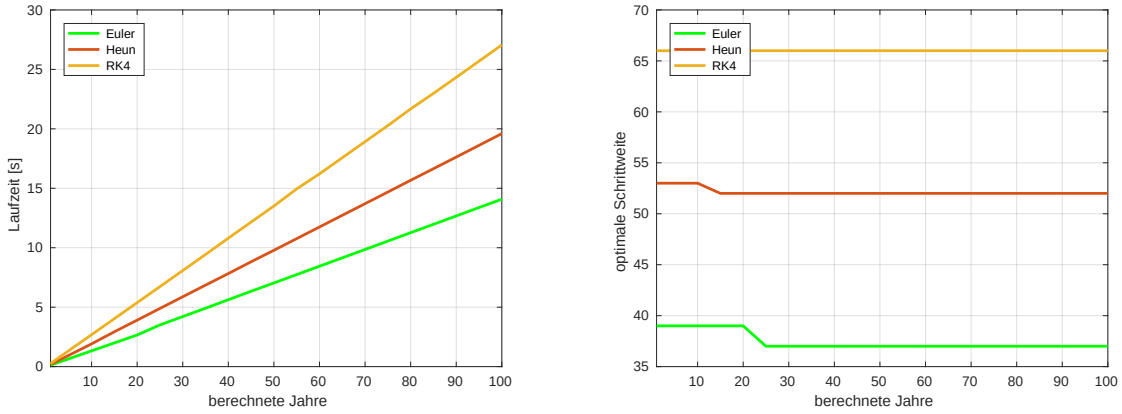
Die Nährstoffaufnahme weicht im Vergleich zu der Mortalität und Diffusion geringer von den Vergleichswerten ab. Dabei fällt diese allerdings zuerst auf einen geringeren Wert (zwischen 11:50 und 13:00 Uhr), um anschließend wieder anzusteigen und eine höhere Nährstoffaufnahme als Euler und RK4 zu aufzuweisen (13:00 bis 13:50 Uhr).

Damit lässt sich schlussfolgern, dass die Abweichung durch die Nährstoffaufnahme verursacht wird, denn diese erreicht zuerst das lokale Minimum und erst danach erreichen Diffusion und Mortalität ihr lokales Maximum. Durch die steigende Nährstoffaufnahme sinken die Diffusion und Mortalität gegen 13 Uhr wieder ab und führen dadurch zu einer geringeren Phytoplanktonkonzentration, die wiederum zu einer Abnahme der Nährstoffaufnahme gegen 13:15 Uhr führt. Gegen 13:45 Uhr ist das Gleichgewicht der Prozesse wieder hergestellt. Der beschriebene Ablauf führt zu der in Abbildung 4.8 sichtbaren Abweichung. Da die Abweichungen zum einen erst zur Mitte des Jahres (siehe Abb. 4.9) und zum anderen um die Mittagszeit (siehe Abb. 4.10) auftreten, lässt sich vermuten, dass das Sonnenlicht eine große Rolle spielt, da zu diesen Zeiten die höchste Einstrahlung erreicht wird. Bei der Berechnung der Nährstoffaufnahme kann es daher zu großen Unterschieden in den Zwischenschritten (bei Heun und RK4) kommen und schließlich zu größeren Abweichungen der Approximationen nach Heun und RK4 gegenüber Euler. Im hier gezeigten Fall ist die Folge eine Abnahme der Nährstoffaufnahme, welche eine geringere Phytoplanktonkonzentration bewirkt und sich wiederum auf die Diffusion und die Mortalität auswirkt. Die verhältnismäßig großen Prozessraten (siehe Tabelle 4.6) unterstützen die Entstehung solcher Abweichungen zusätzlich. Im Rahmen dieser Bachelorarbeit konnte diese Vermutung allerdings nicht verifiziert werden.

Abbildung 4.12 zeigt abschließend den Zusammenhang zwischen der Anzahl der berechneten Jahre und der Laufzeit des NPD-Modells (Abb. 4.12a) bzw. der optimalen Schrittweite (Abb. 4.12b) der Verfahren. Für diese wurden die optimalen Schrittweiten im Fünf-Jahres-Takt (1, 5, 10, 15, ..., 100 Jahre) berechnet und die gemessenen Laufzeiten gemittelt.

Die Laufzeiten steigen nahezu linear an, wobei die minimalen Krümmungen des Heun- und Euler-Verfahrens (nach 15 bzw. 25 Jahren) durch eine Änderung der optimalen Schrittweite verursacht wurde (siehe Abb. 4.12b). Die geringeren optimalen Schrittweiten

entstehen dadurch, dass die Phytoplanktonkonzentration am Anfang bzw. Ende eines Tages mit der Anzahl berechneter Jahre minimal geringer und das Epsilon unterschritten wird. Mit der Abbildung 4.12a zeigt sich, dass für dieses Modell unter Nutzung der oben genannten Parameter zu keinem Zeitpunkt das Heun- oder das RK4-Verfahren schneller als das Euler-Verfahren sind. Ihre Laufzeiten steigen sogar sichtbar schneller an, da das Verhältnis von Schrittweite zum zeitlichen Aufwand der Berechnung der Zwischenschritte zu klein ist. Um eine bessere Laufzeit erzielen zu können, müsste das Heun-Verfahren eine ca. doppelt so große Schrittweite und das RK4-Verfahren eine viermal so große wie das Euler-Verfahren haben.



(a) Laufzeit gegen Anzahl berechneter Jahre. (b) Optimale Schrittweiten gegen Anzahl berechneter Jahre.

Abbildung 4.12: Zusammenhang zwischen der Anzahl berechneter Jahre und der Laufzeit bzw. der optimalen Schrittweite des NPD-Modells.

4.4 Bewertung

Im Abschnitt *Laufzeit- und Genauigkeitsvergleich* wurde unter anderem der Genauigkeitsunterschied der Verfahren gezeigt. Diese Unterschiede konnten zur Reduzierung der zu berechnenden Schritte verwendet werden. Verfahren mit höherer Genauigkeit mussten dadurch weniger Schritte als solche mit geringerer Genauigkeit berechnen. Dadurch konnten die Laufzeiten aufwendigerer Verfahren – unter Einhaltung einer maximalen Abweichung von einem Prozent zu der analytischen Lösung – so stark verringert werden, dass sie die Berechnung schneller als das Euler-Verfahren durchführen konnten. Auch durch die dynamische Schrittweitensteuerung konnten sich die Verfahren nach Heun, AB2 und RK4 gegenüber dem Euler-Verfahren behaupten. Besonders das RK4-Verfahren stach durch seine hohe Genauigkeit heraus, wodurch sich die Schrittweite sehr hoch wählen ließ, was wiederum zu einer guten Laufzeit führte (bei der *eCos*-DGL knapp viermal so schnell). Auch das Heun-Verfahren schnitt sehr gut ab und brachte einen guten Leistungsgewinn (*eCos*-DGL: ca. dreimal so schnell); bei etwa halb so vielen Schritten

wie das RK4-Verfahren konnte es sogar mit diesem konkurrieren. Das AB2-Verfahren stellte zwar in den Analysen ebenfalls eine Verbesserung zu Euler dar, war aber im Vergleich zu den anderen beiden Verfahren sichtbar langsamer und benötigte bei der Schrittweitensteuerung mehr Halbierungen. Damit kann geschlussfolgert werden, dass sich eine Schrittweitensteuerung besonders dann eignet, wenn eine DGL große Steigungen (positiv und negativ) besitzt (beispielsweise die genutzte *eCos*-DGL). In diesem Fall profitieren gerade genaue Verfahren von einer hohen Schrittweite und können trotzdem bei den Extremstellen mit einer niedrigeren Schrittweite gut approximieren. Durch die vergleichsweise schlechte Leistung des AB2-Verfahrens wurde dieses nicht im NPD-Modell implementiert.

Trotz der guten Laufzeiten vom Heun- und RK4-Verfahren in den anfänglichen Analysen, konnten sich diese Verfahren in der Berechnung des NPD-Modells nicht behaupten. Dies hat die Ursache, dass die berechneten optimalen Schrittweiten noch immer zu nahe an der des Euler-Verfahrens liegen. Da der Vorteil vom Heun- und RK4-Verfahren gerade in der höheren Genauigkeit liegt, ergeben sich in diesem Fall durch die aufwendigere Berechnung gegenüber Euler schlechtere Laufzeiten. Dass der Vorteil der Genauigkeit in diesem Modell nicht auffallend, sondern sogar entgegen der Erwartungen ist, liegt an einer ungewöhnlichen Abweichung des Aufnahmeprozesses des Phytoplanktons, die eine deutliche Änderung der Diffusion und der Mortalität verursacht. Die damit sinkende Phytoplanktonkonzentration beeinflusst in weiteren Iterationen ebenfalls die Nährstoff- und Detrituskonzentration den Berechnungsformeln entsprechend. Ursache der Abweichungen ist vermutlich die Sonneneinstrahlung und die verhältnismäßig großen Prozessraten. Da damit eine große Abweichung zu dem erwarteten Ergebnis besteht, können das Heun- und RK4-Verfahren nicht mit dem Euler-Verfahren konkurrieren. Die Abweichungen zeigen aber, dass das genutzte Kriterium zur Bestimmung der optimalen Schrittweiten ungenügend ist und um andere Kriterien ergänzt werden sollte. Da die Laufzeiten trotz höherer Schrittweiten beim Heun- und RK4-Verfahren langsamer als beim Euler-Verfahren waren und die gezeigten Abweichungen auftraten, stellen diese für das Modell keine effizientere Möglichkeit zur Berechnung dar. Die Laufzeiten sind dabei dem höheren Aufwand der Berechnung geschuldet, der durch die Zwischenschritte des Heun- und RK4-Verfahren entsteht.

Zusammenfassung

Die Leistungsanalysen haben gezeigt, dass alle getesteten Verfahren unter Nutzung ihres Genauigkeitvorteils schneller als das Euler-Verfahren sein können. Das Heun-Verfahren muss dafür eine ca. doppelt so große Schrittweite haben, das RK4-Verfahren eine etwa viermal so große und das AB2-Verfahren das 1,5-fache. Hinsichtlich der Auswahl des bestgeeigneten Verfahrens muss im Vorfeld allerdings geklärt werden, wie die zu lösende DGL beschaffen ist. Verfügt die Lösung der DGL über schnell ansteigende oder fallende Maxima, muss ein genaueres Verfahren nicht immer einen Vorteil gegenüber einem Verfahren mit weniger Genauigkeit erzielen. Dies hat den Grund, dass die Berechnung der

Extremstellen eine kleine Schrittweite voraussetzt, da ansonsten zu große Abweichungen zur analytischen Lösung entstehen können. Ist keine dynamische Schrittweitensteuerung vorhanden, muss die Schrittweite entsprechend gering gewählt werden, wodurch der Aufwand eines Verfahrens die damit erzielte Genauigkeit überwiegen kann.

5 Abschluss

In diesem Kapitel werden verwandte Arbeiten und ihre Resultate vorgestellt. Außerdem wird ein Fazit zur Nutzung von DGL-Lösungsalgorithmen höherer Ordnung gegenüber solchen mit geringerer Ordnung gegeben und auf Möglichkeiten zur weiteren Analyse eingegangen.

5.1 Verwandte Arbeiten

In der Bachelorarbeit *Numerical Methods For Solution of Differential Equations* [Rit13] wurden unter anderem fünf numerische Lösungsverfahren implementiert und die Laufzeit bei geringem (10^{-6}) sowie hohem erlaubten Fehler (10^{-3}) für einen Fed-Batch-Prozess (z. B. [Wik16b]) gemessen. Dabei wurden die Verfahren einmal mit adaptiver Schrittweitensteuerung ausgeführt und dreimal mit unterschiedlich großen konstanten Schrittweiten. Zwei der verwendeten Verfahren waren das Euler- und das RK4-Verfahren, bei den anderen handelte es sich um Verfahren mit eigener Abweichungsbestimmung, die adaptiv die Schrittweiten während der Laufzeit bei Bedarf anpassten. Von diesen hatten zwei die Ordnung fünf (Runge-Kutta-Fehlberg und Dormand-Prince54) und waren damit eine Ordnung höher als RK4; Das dritte Verfahren hatte eine Ordnung von drei (ESDIRK23). Bei der Durchführung mit geringem erlaubten Fehler und adaptiver Schrittweitensteuerung war das RK4-Verfahren ca. 8 Mal, das Runge-Kutta-Fehlberg ca. 12 und das Dormand-Prince54 10 Mal schneller als das Euler-Verfahren. Das ESDIRK23-Verfahren war hingegen sogar mehr als 5 Mal langsamer als das Euler-Verfahren, was mit einem Bestandteil der Implementierung zusammen hing. Bei der Durchführung mit konstanten Schrittweiten und ohne adaptiver Schrittweitensteuerung bei allen Verfahren, stellte sich das Euler-Verfahren als schnellstes Verfahren heraus. Am zweitschnellsten war das RK4-Verfahren, gefolgt vom Runge-Kutta-Fehlberg-Verfahren, dann Dormand-Prince54-Verfahren und zum Schluss vom ESDIRK23.

Die Laufzeiten mit adaptiver Schrittweitensteuerung bei hohem erlaubten Fehler, ähnelten bei RK4, Runge-Kutta-Fehlberg und Dormand-Prince54 denen bei geringem erlaubten Fehler. Das Euler- und ESDIRK23-Verfahren waren hingegen deutlich schneller als zuvor. Damit konnte das Euler-Verfahren die anderen Verfahren überholen und die Lösung am schnellsten berechnen. Das ESDIRK23-Verfahren stellte weiterhin das langsamste Verfahren dar.

Da bei drei von vier Analysen mit geringem bzw. hohem erlaubten Fehler einheitliche Schrittweiten für alle Verfahren verwendet wurden, können die Laufzeiten schwer mit den Resultaten dieser Bachelorarbeit verglichen werden. In dieser Arbeit wurden Schrittweiten abhängig von einer gesetzten Abweichungsgrenze bzw. einer Abfrage auf

negative Ergebniswerte hin bestimmt und sind damit für jedes Verfahren unterschiedlich. Die Ergebnisse der Bachelorarbeit [Rit13] stimmen aber im Punkt der geringen Genauigkeiten mit denen dieser Arbeit überein. Wenn keine genaue Approximation, sondern nur eine grobe Lösung notwendig ist, ist das Euler-Verfahren in der Regel am schnellsten, da der Berechnungsaufwand gering ist. Aus den Ergebnissen der hohen Genauigkeit mit adaptiver Schrittweitensteuerung geht hervor, dass das RK4-Verfahren und andere Verfahren höherer Ordnung bei diesen meist schneller Ergebnisse erzielen als das Euler-Verfahren. Dabei ist allerdings auch zu beachten, dass Lösungsverfahren nur dann genauer und damit auch schneller als Verfahren geringerer Ordnung sind, wenn ihre Ordnung mindestens genauso hoch ist, wie die des zu lösenden Problems.

In einem Vergleich numerischer Lösungsverfahren auf der Webseite `http://beltoforion.de/article.php?a=runge-kutta_vs_euler&hl=en` [Ber16] werden mehrere Runge-Kutta- und Adams-Bashforth-Verfahren gegenüber gestellt. Dabei wird die Berechnung gedämpfter harmonischer Schwingungen (z. B. [Bib16]) als Problem verwendet und für alle Verfahren der Berechnungsfehler/Abweichung gegen die Schrittweite aufgetragen. Anschließend werden feste Schrittweiten für jedes Verfahren gewählt und Laufzeiten bestimmt. Am schnellsten ist das Runge-Kutta-Verfahren fünfter Ordnung (RK-Verfahren der höchsten verwendeten Ordnung), mit einem kleinen Geschwindigkeitsvorteil gegenüber dem RK4-Verfahren. Das Adams-Bashforth-Verfahren fünfter Ordnung (höchste Ordnung der verwendeten AB-Verfahren) ist das drittschnellste, sollte aber nach [Ber16] bei einem komplizierteren Problem schneller als die RK-Verfahren sein.

Die Ergebnisse lassen sich gut mit denen dieser Arbeit vergleichen und zeigen ebenfalls, dass unter Einhaltung eines maximalen Fehlers Verfahren höherer Ordnung schneller als solche mit geringerer sind. Auch die geringe Laufzeit der AB-Verfahren lässt sich übertragen, da die in dieser Arbeit verwendeten DGLs ebenfalls einen geringen Berechnungsaufwand erforderten.

5.2 Fazit

Der große Vorteil der Lösungsverfahren von DGL mit hoher Ordnung ist die gelieferte Genauigkeit. Durch diese kann die Schrittweite erhöht werden, die angibt, in welchen Abständen die nächste Approximation berechnet werden soll. Dadurch kann ein aufwendigeres Verfahren die Lösung einer DGL schneller als ein weniger aufwendiges berechnen. Auch eine adaptive Schrittweitensteuerung kann für eine höhere Effizienz von Lösungsverfahren höherer Ordnung sorgen. Dies ist besonders dann empfehlenswert, wenn häufig signifikante Änderungen in der Lösung der Differentialgleichung auftreten und damit eine kleine Schrittweite vorausgesetzt wird. Im Falle einer zu großen Abweichung oder anderen Kriterien kann dann die Schrittweite weiter verringert und im Anschluss wieder erhöht werden. Der Nachteil der Steuerung ist nur, dass Kriterien erstellt werden müssen, damit im richtigen Moment eine Verkleinerung der Schrittweite erfolgt. Diese Abfragen können aber ebenfalls zu einer Erhöhung der Laufzeit führen. Aus diesem Grund muss die Anwendung (z. B. das NPD-Modell) im Vorfeld analysiert und getestet werden. Gerade

wenn keine Schrittweitensteuerung implementiert ist, muss vor Nutzung überprüft werden, ob die höhere Genauigkeit einen Vorteil gegenüber Verfahren mit geringerer Genauigkeit ermöglicht oder ob diese durch den gesteigerten Aufwand die Laufzeit sogar erhöht.

Die Schlussfolgerung dieser Bachelorarbeit ist, dass sich die Implementierung eines der vorgestellten DGL-Lösungsverfahren in dem NPD-Modell nicht lohnt. Dies hat den Grund, dass unter Verwendung der ermittelten optimalen Schrittweiten mit dem Heun- und RK4-Verfahren deutliche Abweichungen zur Vergleichslösung entstehen. Eine Anpassung der Kriterien zur Bestimmung der optimalen Schrittweite würde also vermutlich eine Verringerung der optimalen Schrittweite für das Heun- und RK4-Verfahren bewirken und somit eine weitere Erhöhung der Berechnungsdauer zur Folge haben. Allgemein kann allerdings gesagt werden, dass besonders bei DGL ohne besonders ausgeprägte Extremstellen eine Verbesserung der Laufzeit durch Verwendung von Verfahren höherer Ordnung erreicht werden kann. Für das NPD-Modell könnte möglicherweise unter Verwendung einer gesteuerten Schrittweite eine Leistungssteigerung erreicht werden, dies setzt aber weitere Analysen, insbesondere der Fehlerursache, voraus.

Die Ergebnisse dieser Bachelorarbeit legen nahe, dass eine Kombination aus Verfahren höherer Ordnung und einer adaptiven Schrittweitensteuerung mit dem Kriterium einer maximalen Abweichung für Modelle zu verwenden ist. Dieses kann beispielsweise durch das Runge-Kutta-Fehlberg-Verfahren ermöglicht werden. Dabei wird zusätzlich zu den vier Schritten des RK4-Verfahrens ein fünfter Schritt berechnet. Anschließend wird ein nächster Punkt durch die vier regulären Schritte und ein weiterer aus allen fünf Schritten bestimmt. Ist die Differenz der Punkte größer als eine maximale Abweichung, wird die Schrittweite halbiert. Dieses Verfahren hat den Vorteil, dass nur ein weiterer Schritt und kein zusätzliches Verfahren höherer Ordnung berechnet werden muss, um auf Abweichungen zu überprüfen und die Schrittweite zu halbieren.

5.3 Ausblick

Es gibt noch einige Möglichkeiten zur Verbesserung der Leistung, die es wert sind, weiter nachgegangen zu werden:

So wurde die Schrittweitensteuerung in dieser Arbeit nur wenig behandelt, wird aber im Allgemeinen häufig verwendet. Dazu werden von mehreren Autoren Vorschläge für Kriterien gemacht. Mehr dazu ist zum Beispiel im Buch *Numerik gewöhnlicher Differentialgleichungen: Anfangs- und Randwertprobleme* [Her04, S. 65-71] zu finden.

Auch kann überprüft werden, ob deutlich höhere Ordnungen der hier verwendeten Verfahren zu einer schnelleren Berechnung führen. Diese könnten zu einer besseren Approximation von Extrempunkten führen oder helfen die Abweichungen im NPD-Modell durch das Heun- und RK4-Verfahren zu vermeiden, wodurch anzunehmen ist, dass auch die Schrittweiten höher gewählt werden könnten. In Kombination mit einer Schrittweitensteuerung dürften so vermutlich bessere Laufzeiten erzielt werden können.

Ebenso könnte ein weiteres Mehrschrittverfahren implementiert werden, da diese deutlich weniger Funktionsauswertungen als Einschrittverfahren gleicher Ordnung benötigen.

In dieser Arbeit hat das AB2-Verfahren zwar nur geringe Leistungsverbesserungen ermöglicht, verwendet wurde allerdings eine niedrige Ordnung. Unter Nutzung höherer Ordnungen kann die Schrittweite erhöht und damit die Laufzeit deutlich verbessert werden [Ber16]. Hierbei könnte ein Blick auf die Verfahren nach Adams-Bashforth-Moulton sinnvoll sein, die eine höhere Stabilität als die reinen Adams-Bashforth-Verfahren aufweisen. Dabei werden nicht nur vorherige Punkte analysiert, sondern auch ein nächster Punkt berechnet, der anschließend in die Berechnung mit einfließt und diese korrigiert [ZC09, S. 350-353].

Literaturverzeichnis

[Bac85] Backhaus, J.: A three-dimensional model for the simulation of shelf sea dynamics. In: *Deutsche Hydrografische Zeitschrift* 38 (1985), S. 165–187. http://dx.doi.org/10.1007/BF02328975. – DOI 10.1007/BF02328975

[Ber16] Berg, I.: *Über die Genauigkeit numerischer Integrationsverfahren.* http://beltoforion.de/article.php?a=runge-kutta_vs_euler&hl=de&s=idAccuracy#idAccuracy, 2016. – [Online; Stand 08. August 2016]

[Bib16] Bibliographisches Institut GmbH: *Gedämpfte harmonische Schwingungen.* https://www.lernhelfer.de/schuelerlexikon/physik-abitur/artikel/gedaempfte-harmonische-schwingungen, 2016. – [Online; Stand 29. August 2016]

[But08] Butcher, J. C.: *Numerical Methods for Ordinary Differential Equations.* 2. Auflage. Chichester : John Wiley & Sons, 2008

[Den16] Denning Research Group: *Sunshine on a Perfectly Cloudless Day.* http://biocycle.atmos.colostate.edu/shiny/solar/, 2016. – [Online; Stand 01. August 2016]

[Deu16] Deutsche Klimarechenzentrum GmbH: *DKRZ - Deutsches Klimarechenzentrum.* https://www.dkrz.de/, 2016. – [Online; Stand 20. Mai 2016]

[Ear16] Earth System Research Laboratory: *Solar Calculation Details.* http://www.esrl.noaa.gov/gmd/grad/solcalc/calcdetails.html, 2016. – [Online; Stand 01. August 2016]

[FN14] Fennel, W. ; Neumann, T.: *Introduction to the Modelling of Marine Ecosystems.* 2. Auflage. Oxford : Elsevier Ltd, 2014

[GGK+16] Grosse, F. ; Greenwood, N. ; Kreus, M. ; Lenhart, H.-J. ; Machoczek, D. ; Pätsch, J. ; Salt, L. ; Thomas, H.: Looking beyond stratification: a model-based analysis of the biological drivers of oxygen deficiency in the North Sea. In: *Biogeosciences* 13 (2016), S. 2511–2535. http://dx.doi.org/10.5194/bg-13-2511-2016. – DOI 10.5194/bg–13–2511–2016

[HB16] Honsberg, C. ; Bowden, S.: *Calculation of Solar Insolation*. http://www.pveducation.org/pvcdrom/calculation-of-solar-insolation, 2016. – [Online; Stand 01. August 2016]

[Her04] Hermann, M.: *Numerik gewöhnlicher Differentialgleichungen: Anfangs- und Randwertprobleme*. 1. Auflage. Oldenbourg : De Gruyter, 2004

[LPMK12] Lorkowski, I. ; Pätsch, J. ; Moll, A. ; Kühn, W.: Interannual variability of carbon fluxes in the North Sea from 1970 to 2006 - Competing effects of abiotic and biotic drivers on the gas-exchange of CO_2. In: *Estuar. Coast. Shelf* 100 (2012), S. 38–57. http://dx.doi.org/10.1016/j.ecss.2011.11.037. – DOI 10.1016/j.ecss.2011.11.037

[MF04] Mathews, J. H. ; Fink, K. K.: *Numerical Methods Using Matlab*. 4. Auflage. New Jersey : Prentice Hall, 2004

[Obe08] Oberle, H. J.: *Numerik gewöhnlicher Differentialgleichungen. Skript zur Vorlesung Wintersemester 2008/09*. http://www.math.uni-hamburg.de/home/oberle/skripte/num-gew-dgln/numerik-dgl.pdf, 2008. – [Online; Stand 25. Juli 2016]

[PK08] Pätsch, J. ; Kühn, W.: Nitrogen and carbon cycling in the North Sea and exchange with the North Atlantic – a model study. Part I. Nitrogen budget and fluxes. In: *Cont. Shelf Res.* 28 (2008), S. 767–787. http://dx.doi.org/10.1016/j.csr.2007.12.013. – DOI 10.1016/j.csr.2007.12.013

[Pla00] Plato, R.: *Numerische Mathematik kompakt: Grundlagenwissen für Studium und Praxis*. 1. Auflage. Braunschweig/Wiesbaden : Vieweg Verlagsgesellschaft, 2000

[Poh91] Pohlmann, T.: *Untersuchung hydro- und thermodynamischer Prozesse in der Nordsee mit einem dreidimensionalen numerischem Modell, Berichte aus dem Zentrum für Meeres- und Klimaforschung*. Zentrum für Meeres- und Klimaforschung der Universität Hamburg, 1991 (Berichte aus dem Zentrum für Meeres- und Klimaforschung, Reihe B)

[Poh96] Pohlmann, T.: Predicting the thermocline in a circulation model of the North Sea – Part I: model description, calibration and verification. In: *Cont. Shelf Res.* 16 (1996), S. 131–146. http://dx.doi.org/10.1016/0278-4343(95)90885-S. – DOI 10.1016/0278–4343(95)90885–S

[Rit13] Ritschel, T.: *Numerical Methods For Solution of Differential Equations*, Technical University of Denmark, Bachelorarbeit, 2013

[Vos14] Voss, H.: *Numerische Simulation*. https://www.mat.tuhh.de/lehre/material/num_sim/kap3.pdf, 2014. – [Online; Stand 08. August 2016]

[Wik16a] Wikipedia: *Differentialgleichung — Wikipedia, Die freie Enzyklopädie*. https://de.wikipedia.org/w/index.php?title=Differentialgleichung&oldid=154375995, 2016. – [Online; Stand 14. Juni 2016]

[Wik16b] Wikipedia: *Fed-Batch-Prozess — Wikipedia, Die freie Enzyklopädie*. https://de.wikipedia.org/w/index.php?title=Fed-Batch-Prozess&oldid=146968653, 2016. – [Online; Stand 29. August 2016]

[Wik16c] Wikipedia: *Fick's laws of diffusion — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Fick%27s_laws_of_diffusion&oldid=729412162, 2016. – [Online; Stand 13. August 2016]

[Wik16d] Wikipedia: *Heun-Verfahren — Wikipedia, Die freie Enzyklopädie*. https://de.wikipedia.org/w/index.php?title=Heun-Verfahren&oldid=154421232, 2016. – [Online; Stand 8. Juni 2016]

[Wik16e] Wikipedia: *Linear multistep method — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Linear_multistep_method&oldid=710225648, 2016. – [Online; Stand 14. Juni 2016]

[Wik16f] Wikipedia: *Navier-Stokes-Gleichungen — Wikipedia, Die freie Enzyklopädie*. https://de.wikipedia.org/w/index.php?title=Navier-Stokes-Gleichungen&oldid=156903720, 2016. – [Online; Stand 12. August 2016]

[Win16] Winkler, F. S.: *Aufgabe 5.1 – implizite und explizite Differentialgleichungen*. http://me-lrt.de/implizit-explizit-differentialgleichung-ordnung-auflosen, 2016. – [Online; Stand 30. Juni 2016]

[Wis14] Wissenschaftsrat: *Bedeutung und Weiterentwicklung von Simulation in der Wissenschaft* . http://www.wissenschaftsrat.de/download/archiv/4032-14.pdf, 2014. – [Online; Stand 12. August 2016]

[Wol16] Wolfram Alpha: *Wolfram|Alpha: Computational Knowledge Engine*. https://www.wolframalpha.com/, 2016. – [Online; Stand 19. Juni 2016]

[ZC09] Zill, D. G. ; Cullen, M. R.: *Differential Equations with Boundary-Value Problems*. 7. Auflage. Belmont : Brooks/Cole, 2009

Abbildungsverzeichnis

Listingverzeichnis

Tabellenverzeichnis

Anhänge

A CD-Übersicht

Übersicht der Daten, die auf der CD enthalten sind:

BachelorArbeit.pdf Das PDF der Bachelorarbeit

Messungen/Leistungsanalyse/ Tabellen aller Messungen der Leistungsanalysen, Textdateien mit Angaben der verwendeten Parameter

Messungen/BoxModel/ Tabellen aller Messungen im NPD-Modells, Textdateien mit Angaben der verwendeten Parameter

Plots/Leistungsanalyse/ Auftragungen der allgemeinen Leistungs- und Genauigkeitsanalysen

Plots/BoxModel/ Auftragungen der Analysen des NPD-Modells

Programme/Leistungsanalyse/ Alle geschriebenen Programme zur Leistungsanalyse

Programme/BoxModel/ Alle geschriebenen Programme für die Analyse im NPD-Modell

B Subroutine zur Berechnung des Sonnenlichts

Die Berechnung des Sonnenlichtes aus dem HAMSOM-Modell (Universität Hamburg, Institut für Meereskunde) ist im folgenden Listing dargestellt.

```
1  subroutine solrad(tm, rlat, qs)
2   !
3   ! Argument list
4   !
5   !   tm       input  time in hours TM=0 at midnight January
        ↪ 1st
6   !   rlat     input  latitude  positive N
7   !   qs       output solar radiation at the sea surface
8   !
9   ! Variable list
10  !
11  !   pi         3.1415926536
12  !   dg2rd      conversion factor from degrees to radians
13  !   decln      maximum declination angle of the earth
14  !   w0         angular frequency of one year period
15  !   w1         angular frequency of one day period
16  !   sc         solar constant 1367.0
17  !   d          declination angle at given time
18  !   snH        sine of altitude angle of the sun
19  !   h1         altitude angle of the sun
20  !   q          solar radiation without absorption
21  !   b1         transmission coefficient
22  !
23  implicit none
24  !
25  !     Subroutine arguments.
26  !
27  real(8)  tm, qs, rlat
28  !
29  !     Local variables.
30  !
31  real(8) b1,d,dg2rd,decln,pi,q,sc,snh,w0,w1
```

```
32    !
33    pi = 3.1415926536e0
34    dg2rd = pi/180.0e0
35    decln = 23.5e0*dg2rd
36    w0 = 2.0e0*pi/(365.24e0*24.0e0)
37    w1 = 2.0e0*pi/24.0e0
38    sc = 1367.0e0
39    !
40    !     Calculate sine of the angle of the sun...
41    !     above the horizon, snh
42    !     d is the declination angle...
43    !     June 21st is the 171st day after tm=0
44    !
45    d = decln*cos(w0*tm - 2.950e0)
46    snh = -cos(rlat*dg2rd)*cos(d)*cos(w1*tm) +
        ↪ sin(rlat*dg2rd)*sin(d)
47    snh = min(1.0e0,snh)
48    snh = max(0.0e0,snh)
49    !
50    q = sc*snh
51    !     h1 = asin(snh)
52    !
53    !     Take into account the effect of atmospheric
        ↪ absorption
54    !     Defant 1961 suggests i(z)=i0*exp(-t*a/sin(h1))
55    !     where  t       turbidity factor (unspecified)
56    !            a       0.128 to 0.054 times -ln(sin(h1))
57    !            b1=exp(-t*a)  transmission coefficient
        ↪ between 0.6 and 0.7
58    !     Comparison with Bunker & Goldsmith averages gives
        ↪ b1=0.76 or a=0.274
59    !
60    b1 = 0.76e0
61    qs = (q*b1)/1000
62    !
63    return
64   end
```

Listing B.1: Programmabschnitt aus `npd_mod` für die Berechnung des Sonnenlichtes als Einflussfaktor

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit im Studiengang Computing in Science selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel – insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen – benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und die eingereichte schriftliche Fassung der auf dem elektronischen Speichermedium entspricht.

Ich bin damit einverstanden, dass meine Abschlussarbeit in den Bestand der Fachbereichsbibliothek eingestellt wird.

Ort, Datum	Unterschrift