

Annika Fuchs

**Effiziente parallele Verfahren zur Lösung
verteilter, dünnbesetzter Gleichungssysteme
eines nichthydrostatischen Tsunamimodells**

Die vorliegende Arbeit ist die überarbeitete Fassung einer Dissertation, die am Alfred-Wegener-Institut, Helmholtz-Zentrum für Polar- und Meeresforschung in der Sektion Wissenschaftliches Rechnen angefertigt und am 25. Juli 2013 dem Fachbereich 3 (Mathematik und Informatik) der Universität Bremen vorgelegt wurde. Nach der Begutachtung von Prof. Dr. Wolfgang Hiller und Prof. Dr. Alfred Schmidt fand am 17. September 2013 das Promotionskolloquium statt. Die Überarbeitung begründet sich auf einer anderen Darstellung des nichthydrostatischen Druckgradienten in Abschnitt 2.3.2 und somit einer leicht veränderten Diskretisierung, beschrieben in den Abschnitten 3.2 und 3.3 mit zugehörigen Berechnungen im Anhang. Alle Rechenläufe wurden wiederholt und die Abbildungen der Modellergebnisse in den Kapiteln 4 und 10 ausgetauscht. In Abschnitt 10.2.2 wurde mit der überarbeiteten Version der Teil bezüglich der Schwächen des nichthydrostatischen Tsunamimodells bei steilen Bathymetriegradien obsolet. Bei der Skalierungsuntersuchung in Kapitel 9 sind auftretende Abweichungen zur ursprünglichen Version in den Abschnitten 9.2.5, 9.3 und 9.4.2 innerhalb von vier Fußnoten kommentiert.

Inhaltsverzeichnis

1	Einleitung	1
2	Physikalische Modellierung	5
2.1	Tsunamiwellen	5
2.1.1	Entstehung	5
2.1.2	Ausbreitung	6
2.1.3	Erreichen der Küste	6
2.2	Annahmen und Vereinfachungen	7
2.2.1	Dichte	7
2.2.2	Druck	7
2.2.3	Ausbreitungsgeschwindigkeit	8
2.3	Flachwassertheorie	10
2.3.1	Geschwindigkeitskomponenten	10
2.3.2	Druckanteile	11
2.4	Erhaltungsgleichungen	12
2.4.1	Kontinuitätsgleichung	12
2.4.2	Bewegungsgleichungen	12
3	Modelldiskretisierung	15
3.1	TsunAWI	15
3.1.1	Räumliche Diskretisierung	15
3.1.2	Zeitdiskretisierung	18
3.1.3	Überflutung	18
3.1.4	Flachwassergleichungen	19
3.1.5	Anfangs- und Randbedingungen	19
3.2	Korrektur der Geschwindigkeiten	20
3.2.1	Zusätzliche Unbekannte	20
3.2.2	Anfangs- und Randbedingungen	21
3.3	Assemblierung der Matrizen	22
3.3.1	Massenmatrizen	22
3.3.2	Steifigkeitsmatrix und rechte Seite	23
4	Einfluss des nichthydrostatischen Druckterms	25
4.1	Analytischer Testfall: Stehende Welle im Becken	25
4.1.1	Modellaufbau	26
4.1.2	Vergleich der Modellergebnisse	26
4.2	Tankexperiment: Linearer Anstieg zur Küste	28

4.2.1	Modellaufbau	28
4.2.2	Vergleich der Modellergebnisse	30
4.3	Tankexperiment: Überströmung eines Hindernisses	32
4.3.1	Modellaufbau	32
4.3.2	Vergleich der Modellergebnisse	33
5	Rechengitter und Matrixstruktur	37
5.1	Zusammenhänge	37
5.2	Umsortierung	39
5.3	Beschränkung des Rechengebiets	42
5.3.1	Gesamtmatrix mit konstanten Strukturen	43
5.3.2	Teilmatrix mit variablen Strukturen	44
6	Gebietszerlegung	45
6.1	Partitionierung	45
6.1.1	Gebietszerlegung	45
6.1.2	Berechnung und Synchronisation	46
6.1.3	Partitionierungssoftware	47
6.1.4	Gewichtung des Graphs	48
6.2	Das verteilte Gleichungssystem	48
6.2.1	Lokale Umsortierung	50
6.2.2	Gesamt- oder reduziertes System	51
7	Krylov-Unterraumverfahren	53
7.1	Iteratives Lösungsverfahren	53
7.2	Lösungsraum	54
7.3	Basisvektoren	55
7.4	Verfahrensprinzip	56
7.5	GMRES mit Vorkonditionierung	57
8	Präkonditionierungsmodelle	61
8.1	Präkonditionierungsbausteine	61
8.1.1	Einfaches Skalieren	61
8.1.2	Unvollständige LU-Zerlegung	62
8.2	Vom globalen zum lokalen System	65
8.2.1	Block-Jacobi Verfahren	66
8.2.2	Restricted Additive Schwarz Methode	68
8.3	Verschachtelte Präkonditionierungsmethoden	69
8.3.1	Das Schurkomplement	69
8.3.2	Vorkonditionierung der Schursystems	72
9	Effiziente Berechnungsmethoden	75
9.1	Übersicht	75
9.1.1	Ablauf	75
9.1.2	Software	76
9.1.3	Hardware	76

9.1.4	Kenngrößen	77
9.2	Präkonditionierungsmethoden	79
9.2.1	Faktorisierungsansätze	80
9.2.2	Datenstrukturen	81
9.2.3	Vergleich der Methoden zur parallelen Vorkonditionierung	83
9.2.4	Einfluss der Gitternummerierung	87
9.2.5	Vorkonditionierung beim Tsunamiszenario	91
9.3	Beschränkung auf das Teilsystem	96
9.4	Partitionierung	99
9.4.1	Gewichtete Partitionierung	99
9.4.2	Partitionierung in zwei Schritten	101
10	Simulation von Tsunamireignissen	107
10.1	Tōhoku-Szenario	107
10.1.1	Modellaufbau	108
10.1.2	Ankunftszeit und maximale Wellenhöhe	108
10.1.3	Pegelstände	109
10.2	Mentawai-Szenario	111
10.2.1	Pegelstandsaufzeichnungen als Referenz	111
10.2.2	Ankunftszeit und maximale Wellenhöhe	113
10.3	Bourmedès-Zemmouri-Szenario	113
10.3.1	Modellaufbau	114
10.3.2	Ankunftszeit und maximale Wellenhöhe	116
11	Zusammenfassung und Ausblick	119
	Anhang	123
A	Herleitung der verwendeten Gleichungen	123
A.1	Vertikal gemittelte nichthydrostatische Druckgradienten	123
A.2	Flachwassergleichungen	124
A.3	Korrekturterme	124
A.4	Schwache Formulierung der Kontinuitätsgleichung	125
A.5	Assemblierung	126
	Bezeichnungen	129
	Abbildungsverzeichnis	131
	Algorithmenverzeichnis	133
	Tabellenverzeichnis	134
	Literaturverzeichnis	135
	Danksagung	141

1. Einleitung

Als Folge des verheerenden Tsunamis vom 26. Dezember 2004 im Indischen Ozean wurde innerhalb des GITEWS-Projekts [52], [43] ein deutsch-indonesisches Tsunamifrühwarnsystem aufgebaut, bei dem auf eine Datenbank vorbereiteter Szenarien zugegriffen wird [9]. Auch in den Frühwarnzentren der JMA (Japan Meteorological Agency) [22] und NOAA (National Oceanic and Atmospheric Administration) [67] gehen vorausberechnete Tsunamiszenarien in die Warnprodukte mit ein. Die operationell genutzten Tsunamimodelle basieren dabei für gewöhnlich auf den nichtlinearen, vertikal gemittelten Flachwassergleichungen unter der hydrostatischen Annahme, dass sich Auftrieb und Gravitation im Gleichgewicht halten. Die vertikale Geschwindigkeit und der nichthydrostatische Druckanteil werden dabei vernachlässigt. Klassische Beispiele wie die Simulationsprogramme MOST (Method of Splitting Tsunamis) [66] und TUNAMI-N&F (Tohoku Universitys Numerical Analysis Model for Investigation of Near-field and Far-field tsunamis) [31] sind mit dem Finite Differenzen Verfahren umgesetzt. Erreicht die Tsunamiwelle die Küste, so wird der Geltungsbereich des Flachwassermodells verlassen. Mit einer feinen Diskretisierung und detaillierten Geometriedaten bezüglich Bathymetrie und Topographie kann dennoch Überflutung simuliert werden. Mit Hilfe von ineinander verschachtelten Gittern mit unterschiedlich feiner Diskretisierung werden einzelne Küstenabschnitte gezielt hoch aufgelöst. Außerhalb dieser Prioritätsgebiete werden relevante Größen wie die maximale Wellenhöhe und die Ankunftszeit des Tsunamis mit Hilfe von Formeln abgeschätzt, die sich auf die Modellergebnisse in küstennahen Punkten in tiefem Wasser stützen. In neueren Entwicklungen wird auf unstrukturierten Gittern gerechnet, bei denen die unterschiedlichen Skalen im tiefen Ozean und an der Küste ohne Abstufung abgedeckt werden. Die oben genannten Größen können somit an der gesamten Küste direkt bestimmt werden. So zum Beispiel das Finite Volumen Überflutungsmodell AnuGA [45] von Geoscience Australia und der National Australian University. Für das deutsch-indonesische Tsunamifrühwarnsystem wurde am Alfred-Wegener-Institut, Helmholtz-Zentrum für Polar- und Meeresforschung (AWI) das Simulationsprogramm TsunAWI [3], [50] entwickelt. Das vertikal gemittelte, zweidimensionale Flachwassermodell wurde dabei aus dem bestehenden Finite Elemente Ozean Modell (FEOM) [16] abgeleitet, das zur Simulation globaler dreidimensionaler Ozeanzirkulationen dient. Bei der Simulation realer Tsunamiereignisse bieten die Modellergebnisse des Flachwassermodells auch an der Küste oft noch eine gute Übereinstimmung mit aufgezeichneten Messwerten, wie in [28] im Vergleich von TsunAWI und TUNAMI-N3 gezeigt wird. Für [26] wurden Sensitivitätsstudien durchgeführt, die sich mit dem Einfluss von Diskretisierung, Geometriedatenqualität und verwendetem Überflutungsmodell (TsunAWI, AnuGA oder TUNAMI-N3) auseinandersetzen. In Gebieten mit steilen Bathymetriereläufen nahe der Küste, wie sie zum Beispiel im Mittelmeer zu finden

1. Einleitung

sind, können nichthydrostatische Effekte erwartet werden, die das Flachwassermodell nicht auflösen vermag. Um in diesen Gebieten eine höhere Genauigkeit zu erlangen, wird in dieser Arbeit von der hydrostatischen Annahme Abstand genommen, so dass die vertikale Geschwindigkeit und der hydrodynamische Druckanteil ins Modell mit einfließen. Für die numerische Simulation bedeutet das erhöhte Last durch den zusätzlichen Rechenaufwand und den benötigten Speicherplatz. Im Anbetracht des Fluchs der Dimensionen wird hier kein dreidimensionales Modell aufgesetzt. Im Vergleich zum bestehenden hydrostatischen zweidimensionalen Flachwassermodell wäre der Rechenaufwand exorbitant. Deswegen wird hier der Ansatz verfolgt, TsunAWI um ein Modul zu erweitern, welches den Geltungsbereich dieses Simulationsprogramms erweitert, indem der nichthydrostatische Druckanteil in die Tsunamimodellierung mit einfließt. Wird durch eine geschickte Kombination von numerischen Methoden und die effiziente Ausnutzung von Rechenkapazitäten ein komplexes Tsunamiszenario in akzeptabler Zeitspanne mit dem nichthydrostatischen Ansatz berechnet, so lässt sich diese Arbeit gegebenenfalls durch die Erweiterung in ein mehrschichtiges zweidimensionales Modell fortsetzen.

TsunAWI basiert auf der \mathcal{P}^1 - P_{NC}^1 Finite Elemente Diskretisierung [27] für unstrukturierte Gitter mit dem Leapfrog-Zeitschrittverfahren, das eine explizite zeitliche Vorwärtsintegration ermöglicht. In den zugrunde liegenden Gleichungen werden abgesehen vom vorherrschenden Druckgradienten äußere Einflüsse wie Coriolis-, Viskositäts- und Reibungskräfte in den nichtlinearen Flachwassergleichungen berücksichtigt. Zudem ist durch Extrapolation von Modellgrößen [39] ein Überflutungsschema gegeben. Im Druckgradienten wird jedoch nur der hydrostatische Anteil betrachtet. Zur Erweiterung in ein nichthydrostatisches Modell TsunAWI-NH wird die horizontale Geschwindigkeitskomponente als Zwischenergebnis verwendet und in jedem Zeitschritt unter Berücksichtigung des Gradienten des nichthydrostatischen Druckanteils korrigiert [12], [62], [72]. Für diese Modifikation müssen zusätzlich die Größen Vertikalgeschwindigkeit und dynamischer Bodendruck bestimmt werden, wobei die Geschwindigkeitsberechnung zunächst ebenfalls der hydrostatischen Annahme unterliegt und anschließend korrigiert wird. Um die vertikale Geschwindigkeitskomponente explizit berechnen zu können, wird die entsprechende Massenmatrix mittels strikter Diagonalisierung approximiert. Die Bestimmung des Bodendrucks verlangt jedoch, dass in jedem Zeitschritt ein großes, dünnbesetztes Gleichungssystem mit zeitabhängigen Koeffizienten aufgestellt und gelöst wird. Da die Nummerierung von Elementen und Knoten im Rechengitter eng mit der Struktur der Matrix des Gleichungssystems zusammenhängt, kann mit Hilfe einer bewussten Vorsortierung sowohl der Speicherzugriff verbessert, Speicherplatz gespart und die Lösungsalgorithmen des Gleichungssystems beschleunigt werden. Bei der Tsunamisimulation mit Überflutung liegt ein großer Anteil Gitterknoten an Land. Mit einer Beschränkung auf ein variables Rechengebiet kann das Gleichungssystem auf ein Teilsystem reduziert werden. Um externe MPI-parallele Löserbibliotheken einbinden zu können, wurde TsunAWI, dessen parallele Berechnung ursprünglich auf OpenMP-Direktiven beruht, entsprechend umstrukturiert und parallelisiert. Durch eine Gebietszerlegung wird sowohl das Rechengebiet partitioniert, als auch das Gleichungssystem verteilt. Eine geschickte Partitionierung kann dabei erhebliche Verbesserungen bezüglich des gleichmäßigen Lastausgleichs ergeben,

wovon im Besonderen die Lösung des reduzierten Teilsystems profitiert. Ausgehend vom Löserpaket pARMS 3.2 [38] werden verschiedene Präkonditionierungsmethoden für das Krylov-Unterraumverfahren FGMRES mit Neustart [53] untersucht. Nicht unterstützte Ansätze werden dafür implementiert und in das Löserpaket eingebettet. So zum Beispiel ein Modul, welches Datenstrukturen bei Bedarf trotz wechselnder Matrixeinträge wiederverwendet. Auch die Schur+RAS Methode [37] zur parallelen Vorkonditionierung des Gleichungssystems wurde umgesetzt. Dabei operiert der Restricted Additive Schwarz Algorithmus (RAS) [11] auf einem approximierten globalen Schursystem [56].

Für diese Arbeit wird der Finite Elemente Ansatz für das nichthydrostatische, vertikal gemittelte Modell aus [72] in das Tsunamisimulationsprogramm TsunAWI übertragen. Anhand von Standardtestbeispielen wird gezeigt, dass mit der nichthydrostatischen Erweiterung eine Verbesserung des Flachwassermodells erreicht wird. Während das hydrostatische Modell nur bei reinen Flachwasserbewegungen verwendet werden kann, liefert TsunAWI-NH auch in einem Übergangsbereich noch gute Ergebnisse. Brechende Wellen können zwar nach wie vor im zweidimensionalen Modell nicht dargestellt werden, doch ist es mit dem nichthydrostatischen Ansatz möglich, die damit verbundenen dispersiven Effekte gut abzubilden, während die hydrostatisch modellierte Welle eine Sägezahnform mit unnatürlich steiler Wellenfront annimmt. Das Prinzip, parallele Lösungsalgorithmen mittels MPI-Kommunikationsroutinen zwischen den Prozessoreinheiten durchzuführen, wird für diese Arbeit innerhalb des Tsunamimodells umgesetzt, so dass TsunAWI und TsunAWI-NH auch auf Architekturen eingesetzt werden können, deren Prozessoreinheiten nicht auf einen gemeinsamen Speicher zugreifen. Dies ist ein wesentlicher Fortschritt für TsunAWI, da mit der MPI-Version eine höhere Anzahl an Prozessoreinheiten verwendet und die benötigte Laufzeit erheblich verringert werden kann. Falls nicht anders gekennzeichnet ist in dieser Arbeit mit TsunAWI immer die MPI-parallele Version gemeint. Zur Vorprozessierung wird das Rechengebiet mit einem externen Programm in Teilgebiete zerlegt, wobei in dieser Arbeit verschiedene Ansätze untersucht werden, wie trotz heterogenem Rechenaufwand die Rechenlast gleichmäßig auf die Prozessoreinheiten verteilt werden kann. Mittels ausführlicher Untersuchungen von verschiedenen Vorkonditionierungsmethoden und Gittervorsortierungsalgorithmen soll ein grundlegendes Verständnis für die Vor- und Nachteile der numerischen Methoden entwickelt werden, so dass sich diese für die nichthydrostatische Tsunamisimulation mit Überflutung effizient einsetzen lassen. Somit kann der im Verhältnis zum hydrostatischen Flachwassermodell rechen- und speicherintensive nichthydrostatische Ansatz auch für die komplexe Simulation von realen Ereignissen eingesetzt werden. Zum Beispiel wird die Berechnung des Tōhoku-Tsunamis 2011 mit einer Integrationsdauer von dreißig Stunden auf einem globalen Rechengitter durchgeführt. Bei der Simulation von realen Tsunamieignissen werden die Modellergebnisse von TsunAWI und TsunAWI-NH mit gemessenen Referenzwerten verglichen.

Diese Arbeit ist folgendermaßen strukturiert: In Kapitel 2 werden physikalische Grundlagen der Tsunamibewegung erläutert und Annahmen sowie Vereinfachungen der zugrunde liegenden Bewegungsgleichungen getroffen. Die zweidimensionale Modellierung wird mit der Flachwassertheorie begründet und die Erhaltungsgleichungen

1. Einleitung

werden dargestellt. Kapitel 3 befasst sich mit der Diskretisierung und der Berechnung der zusätzlichen Unbekannten. Der Vergleich von hydrostatischen und nicht-hydrostatischen Modellergebnissen mit Referenzdaten bei idealisierten Experimenten unterschiedlicher Komplexität wird in Kapitel 4 dargestellt. Die Wechselbeziehung zwischen dem Rechengebiet und der Matrixstruktur im Gleichungssystem wird in Kapitel 5 zusammen mit der Beschreibung der verwendeten Permutationen zur Gittervorsortierung aufgezeigt. Zudem wird das veränderliche Rechengebiet vorgestellt, das sich aus nassen und direkt angrenzenden trockenen Knoten erschließt. In Kapitel 6 wird die Gebietszerlegung für die parallele Berechnung mit resultierender Verteilung des Gleichungssystems beschrieben. Das genutzte Krylov-Unterraumverfahren zur Lösung des Gleichungssystems wird in Kapitel 7 umrissen, die verwendeten Präkonditionierungsmodelle in Kapitel 8 ausführlicher erklärt. Dabei wird sowohl auf parallele als auch serielle Verfahren und verschachtelte Methoden basierend auf dem approximierten Schursystem eingegangen. Die Untersuchungen der numerischen Verfahren für die nichthydrostatische Erweiterung sind in Kapitel 9 enthalten. Kapitel 10 zeigt Unterschiede zwischen den Ergebnissen von TsunAWI und TsunAWI-NH bei verschiedenen komplexen Simulationen die auf realen Tsunamiereignissen beruhen. Zusammenfassung und Ausblick sind in Kapitel 11 zu finden. Im Anhang A sind die Herleitungen von Gleichungen aufgeschlüsselt, die in den Kapiteln 2 und 3 ohne viele Zwischenschritte dargestellt wurden.

2. Physikalische Modellierung

Nach einer kurzen Einführung in die Bewegungscharakteristik von Tsunamiwellen werden Annahmen definiert, welche die Tsunami-Modellierung vereinfachen. Mit Hilfe der Flachwassertheorie kann das Modell auf zwei Dimensionen mit vertikal gemittelten Werten reduziert werden. Von der üblichen Vernachlässigung des nichthydrostatischen Druckanteils wird hier jedoch Abstand genommen. Durch das Aufstellen von Erhaltungsgleichungen kann dann das Modell beschrieben werden.

2.1. Tsunamiwellen

Die Existenzdauer einer Tsunamiwelle kann in drei Abschnitte unterteilt werden: Entstehung, Ausbreitung und das Erreichen der Küste. Jede dieser drei Phasen birgt charakteristische Verhaltensweisen der Welle, welche hier zusammengefasst dargestellt und in Abbildung 2.1 skizziert sind. Eine genauere Darstellung der vorherrschenden Physik ist in [36] zu finden. Die bekanntesten Tsunami-Beispiele in den letzten Jahren sind der Sumatra-Andaman Tsunami am 26. Dezember 2004 im Indischen Ozean und der Tōhoku-Tsunami im Pazifik vom 11. März 2011.

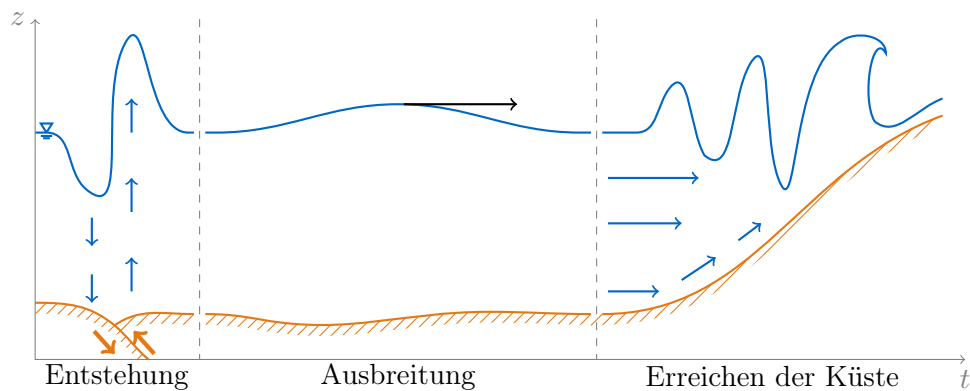


Abbildung 2.1.: Chronologische Folge der Bewegungscharakteristiken eines Tsunamis.

2.1.1. Entstehung

Tsunamiwellen können aus unterschiedlichen Gründen ausgelöst werden: als Folge von Seebeben, Vulkanausbrüchen oder Hangrutschungen unter Wasser, in seltenen Fällen

2. Physikalische Modellierung

auch durch einen Meteoriteneinschlag im Ozean. Zudem gibt es auch meteorologische Tsunamis, welche bei extremen Wetterlagen mit großen Luftdruckschwankungen entstehen können. Die folgende Abhandlung beschränkt sich auf den Fall, dass der Tsunami durch ein submarines Beben angeregt wird. Wenn sich ein Teil des Meeresbodens plötzlich hebt beziehungsweise senkt, wird die darüberliegende Wassersäule mitbewegt. Dies kann eine Störung mit mehreren Metern Auslenkung in vertikaler Richtung bedeuten. Um in den Gleichgewichtszustand zurückzufinden, breitet sich eine Welle horizontal nach allen Seiten aus.

2.1.2. Ausbreitung

Die Fortpflanzung einer Tsunamiwelle wird von den wirkenden Druckunterschieden angetrieben, wobei bei der weitreichenden Bewegung auf dem Ozean auch Corioliskräfte eine Rolle spielen. Auf hoher See ist eine Tsunamiwelle nur schwer erkennbar, da sie zwar eine enorme Wellenlänge, aber nur eine geringe Wellenhöhe aufweist. Sie rast mit einer hohen Ausbreitungsgeschwindigkeit über den Ozean hinweg. Beispielhafte Werte dazu sind Tabelle 2.1 zu entnehmen. Im Gegensatz zu winderzeugten Wellen bewegt sich bei einem Tsunami nicht nur eine dünne Schicht unter der Wasseroberfläche, sondern die gesamte Wassersäule. Da die Wellenlänge verglichen mit der Wassertiefe sehr groß ist, erfüllt ein Tsunami in tiefem Wasser genau die Kriterien, mit denen Flachwasserwellen charakterisiert werden, siehe Abschnitt 2.2.3. In diesem Fall wird die Ausbreitungsgeschwindigkeit fast ausschließlich von der Wassertiefe bestimmt. Jedoch nur fast, denn auch die Wellenlänge spielt eine Rolle und bei fortschreitender Ausbreitung setzen nach einer Distanz von tausend oder mehr Kilometern dispersive Effekte ein, die dazu führen, dass die Tsunamiwelle zu einem Wellenpaket zerfällt [36].

Tabelle 2.1.: Größenordnung charakteristischer Parameter einer Tsunamiwelle [36].

	Wassertiefe	Wellenlänge	Amplitude	Geschwindigkeit
tiefer Ozean	~ 4000 m	~ 100 km	< 1 m	~ 720 km/h

2.1.3. Erreichen der Küste

Beim Zurücklegen weiter Strecken verliert die Tsunamiwelle an Energie, zum einen durch Reibungskräfte, andererseits durch Hindernisse, wie beispielsweise Unterwassergebirge sie darstellen. Bestenfalls genügt dieser Energieverlust, um das anfängliche Zerstörungspotential der Welle abzuschwächen, so dass beim Erreichen der Küste kein Schaden verursacht wird. Leider tritt dieser Fall gerade bei starken Beben nahe der Küste selten ein. Der Anstieg des Meeresgrunds in den Küstengewässern wirkt abbremsend, der Einfluss nichtlinearer Advektion sowie der Viskositäts- und Reibungsterme steigt an. Da die geringe Wassertiefe zu wenig Platz für die ankommenden Wassermassen bietet, türmt sich die Tsunamiwelle auf und es kann zu Überflutungen mit verheerenden Auswirkungen kommen.

2.2. Annahmen und Vereinfachungen

Im weiteren Verlauf werden folgende Notationen verwendet: Das Fluid Wasser wird in vertikaler Richtung durch den undurchlässigen Meeresgrund $-h(x, y)$ und der freien Oberfläche begrenzt, deren Auslenkung aus der Ruhelage mit $\eta(t, x, y)$ dargestellt wird. Die Gesamttiefe wird demnach durch $H = \eta + h$ beschrieben. Während sich Wasserteilchen mit der Geschwindigkeit $\mathbf{v}(t, x, y, z)$ bewegen, breitet sich ein Tsunami der Wellenlänge λ mit der Fortpflanzungsgeschwindigkeit c aus. In Abbildung 2.2 sind diese Größen schematisch dargestellt. Temperaturunterschiede und Salzgehalt bleiben unbeachtet.

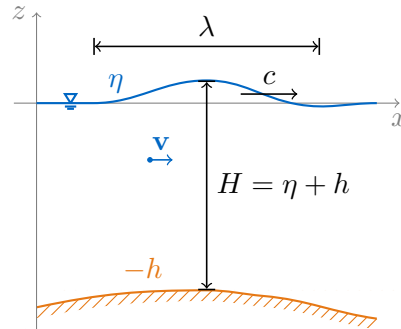


Abbildung 2.2.: Skizze der verwendeten Größen.

2.2.1. Dichte

Bei der Tsunamisimulation wird von einem homogenen, inkompressiblen Fluid ausgegangen. Das bedeutet, dass die Wasserdichte ρ sich weder zeitlich ändert, noch ortsabhängig ist, sondern im gesamten System einen konstanten Wert besitzt:

$$\partial_t \rho = \partial_x \rho = \partial_y \rho = \partial_z \rho = 0. \quad (2.1)$$

Die konstante Dichte ρ taucht im Folgenden nur noch im Skalierungsfaktor $1/\rho$ auf.

2.2.2. Druck

Bei der Betrachtung des über die konstante Dichte ρ skalierten Drucks $p(t, x, y, z)$ kann zwischen dem hydrostatischen Druckanteil p_0 und dem hydrodynamischen oder nichthydrostatischen Druckanteil p' unterschieden werden. Bei Vernachlässigung des Atmosphärendrucks gilt

$$p = p_0 + p'. \quad (2.2)$$

Das Gewicht der Wassersäule, die über einem Punkt liegt, bewirkt den hydrostatischen Druckanteil. Über die Dichte skaliert, kann dieser mit

$$p_0 = g(\eta - z) \quad (2.3)$$

2. Physikalische Modellierung

berechnet werden, wobei g die Erdbeschleunigung darstellt. Die Druckverteilung in vertikaler Richtung besitzt einen linearen Verlauf: Sie verschwindet an der Oberfläche und steigt proportional zur Wassertiefe. Analog dazu wird nun auch für den nichthydrostatischen Druck eine lineare Druckverteilung in z -Richtung angenommen, vergleiche [72].

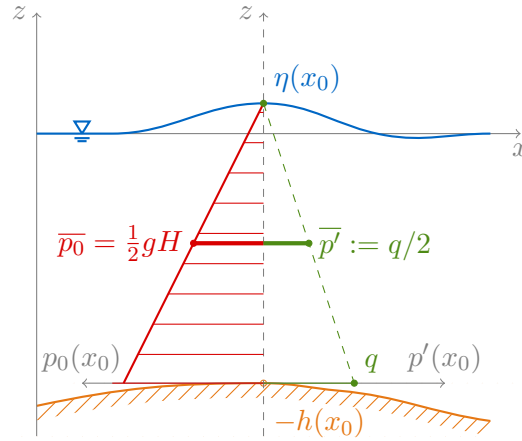


Abbildung 2.3.: Skizze des gegebenen beziehungsweise angenommenen Verlaufs des hydrostatischen und nichthydrostatischen Druckanteils an der Stelle x_0 und die entsprechenden Durchschnittswerte \bar{p}_0 und \bar{p}' .

Da an der Oberfläche sowohl p_0 als auch p' verschwinden müssen, kann mit dieser Annahme p' über den hydrodynamischen Bodendruck $q := p'(-h)$ dargestellt werden:

$$p' := \frac{q}{H}(\eta - z). \quad (2.4)$$

Wird im Folgenden von der hydrostatischen Annahme gesprochen, so ist damit die Vernachlässigung des nichthydrostatischen Druckanteils gemeint. Es wird also angenommen, dass sich Auftrieb und Gravitation im Gleichgewicht halten.

2.2.3. Ausbreitungsgeschwindigkeit

Die Ausbreitungsgeschwindigkeit c einer Wellenbewegung repräsentiert die Geschwindigkeit eines Wellenbergs. Diese hängt von der Wassertiefe H und der Wellenlänge λ ab. Einzelne Fluidteilchen innerhalb des Systems bewegen sich mit kleiner Auslenkung um den Ruhepunkt und geben so die Wellenbewegung an andere Teilchen weiter. Die Gleichung

$$c = \sqrt{\frac{\lambda g}{2\pi} \tanh\left(\frac{2\pi H}{\lambda}\right)} \quad (2.5)$$

liefert eine allgemeine Formel zur Berechnung der Fortpflanzungsgeschwindigkeit. Die Herleitung ist in [1] zu finden.

In Tabelle 2.2 sind die Abhängigkeiten der Fortpflanzungsgeschwindigkeit bezüglich Wellenlänge und Wassertiefe in verschiedenen Fällen aufgelistet. Die Formeln, mit

denen c in sehr tiefem beziehungsweise flachem Wasser abgeschätzt werden kann, sind Grenzfälle der Gleichung (2.5). Im Flachwasser, definiert über das Verhältnis $H/\lambda \ll 1$, kann $\tanh(2\pi H/\lambda) \approx 2\pi H/\lambda$ abgeschätzt werden, in sehr tiefem Wasser konvergiert die Hyperbelfunktion für $H/\lambda \gg 1$ gegen Eins.

Tabelle 2.2.: Ausbreitungsgeschwindigkeit c je nach Verhältnis H/λ .

Klassifikation	H/λ	$c(\lambda)$	$c(H)$	c
sehr tiefes Wasser	$\gg 1$	\times		$c \approx \sqrt{\lambda g/2\pi}$
mäßig tiefes Wasser	≈ 1	\times	\times	(2.5)
flaches Wasser	$\ll 1$		\times	$c \approx \sqrt{gH}$

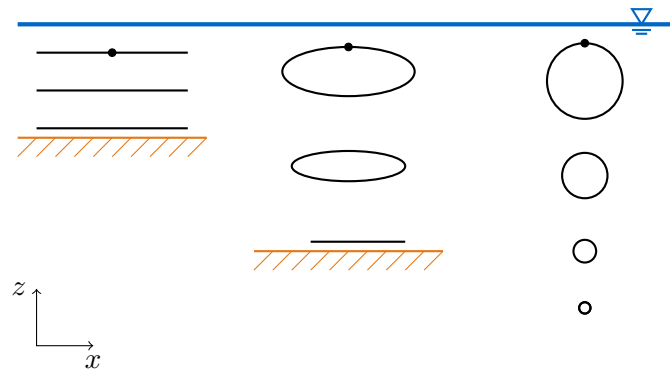


Abbildung 2.4.: Skizze der Bahnkurven in flachem Wasser (links), mäßig tiefem Wasser (mittig) und sehr tiefem Wasser (rechts) [1].

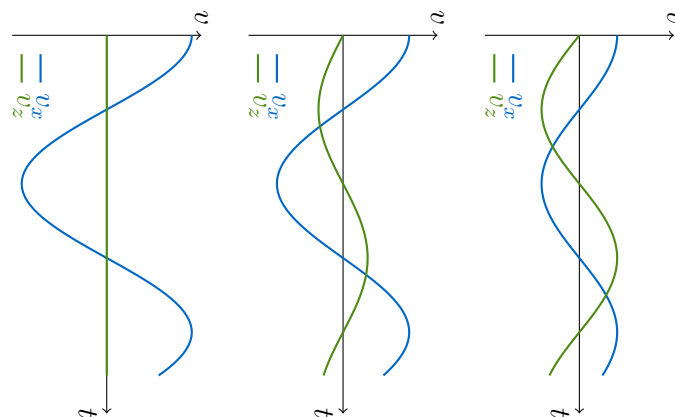


Abbildung 2.5.: Schematische Darstellung der Geschwindigkeitskomponenten entsprechen der jeweils oberen Bahnkurve aus Abbildung 2.4.

2. Physikalische Modellierung

In Abbildung 2.4 sind die Bahnkurven abhängig von der Wassertiefe skizziert [1]. Es ist leicht zu erkennen, dass in flachem Wasser die Bewegung über die gesamte Wassertiefe vergleichbar in horizontaler Richtung erfolgt, während in sehr tiefem Wasser die Bewegung hauptsächlich in der Oberflächenschicht stattfindet und die Bewegung auf Kreisbahnen verläuft. Die zugehörigen Geschwindigkeitskomponenten des Teilchens auf der jeweils obersten Bahnkurve sind in Abbildung 2.5 skizziert. Der Anteil der Vertikalgeschwindigkeit ist im Flachwasser verschwindend gering. In tiefem Wasser hingegen sind horizontale und vertikale Anteile gleichwertig.

Die übliche Klassifizierung gemäß der Wassertiefe ist irreführend, da eine Tsunamiwelle auf hoher See mit dem entsprechenden Verhältnis H/λ genau in die Flachwasserkategorie fällt, während beim Erreichen der Küste die vertikale Geschwindigkeitskomponente an Einfluss gewinnt.

2.3. Flachwassertheorie

Ist bei einer Fluidbewegung die vertikale Bewegung gegenüber der horizontalen verschwindend gering, so ist das grundlegende Kriterium der Flachwassertheorie erfüllt [47]. Bei der Verwendung von Gesamtwassertiefe H und Wellenlänge λ als Referenzwerte für vertikale und horizontale Bewegungen, gilt in diesem Fall das Verhältnis

$$\frac{H}{\lambda} \ll 1. \quad (2.6)$$

Verglichen mit den Werten aus Tabelle 2.1 trifft dies für einen Tsunami in der Ausbreitungsphase zu. Zusammen mit der hydrostatischen Annahme kann mit Hilfe eines zweidimensionalen Modells, basierend auf vertikalen Durchschnittswerten, die tatsächliche Bewegung sehr gut angenähert werden. Die vertikale Geschwindigkeitskomponente und der nichthydrostatische Druckanteil werden dabei vernachlässigt.

Dieser Ansatz liefert eine sehr gute Approximation für die Ausbreitung von Tsunamiwellen, da sich in dieser Phase die Wellen tatsächlich wie Flachwasserwellen verhalten. Erreicht die Welle jedoch Küstenbereiche, so verändert sich das Verhältnis H/λ und der Einfluss der vertikalen Bewegungen steigt. Um in somit kritischen Bereichen genauere Ergebnisse erzielen zu können, wird des Weiteren von der hydrostatischen Annahme Abstand genommen. Die Herangehensweise, die dreidimensionale Fluidbewegung über Mittelwerte auf ein zweidimensionales Modell zu reduzieren, wird jedoch aufgegriffen und das Flachwassermodell wird verwendet, um Zwischenergebnisse zu produzieren. Da die vertikale Geschwindigkeitskomponente eine gesonderte Stellung inne hat, wird im weiteren Verlauf häufig zwischen horizontaler und vertikaler Geschwindigkeit unterschieden.

2.3.1. Geschwindigkeitskomponenten

Beschreibt $\mathbf{v}(t, x, y, z) = (v_x, v_y, v_z)^T$ den Geschwindigkeitsvektor zum Zeitpunkt t an der Position (x, y, z) , so gilt für die Komponenten des gemittelten horizontalen

Geschwindigkeitsvektors $\mathbf{u}(t, x, y) = (u, v)^T$

$$u = \frac{1}{H} \int_{-h}^{\eta} v_x \, dz \quad \text{und} \quad v = \frac{1}{H} \int_{-h}^{\eta} v_y \, dz. \quad (2.7)$$

Kinematische Randbedingungen bestimmen die vertikale Geschwindigkeit an der Oberfläche und am Grund. Diese lauten bei der Verwendung von vertikal gemittelten Geschwindigkeitskomponenten:

$$w_{\eta} = \partial_t \eta + \mathbf{u} \cdot \nabla \eta, \quad (2.8)$$

$$w_{-h} = -\partial_t h - \mathbf{u} \cdot \nabla h. \quad (2.9)$$

Unter Annahme eines linearen Verlaufs in z -Richtung [72] wird die gemittelte vertikale Geschwindigkeitskomponente $w(t, x, y)$ durch

$$w = \frac{1}{2}(w_{\eta} + w_{-h}) \quad (2.10)$$

beschrieben. In den hier behandelten Fällen wird die aus dem Erdbeben resultierende Veränderung des Grunds als Anfangsbedingung behandelt, so dass des Weiteren $\partial_t h \equiv 0$ angenommen werden kann.

2.3.2. Druckanteile

Über die Tiefe gemittelt betragen die Druckanteile p_0 und p' nach (2.3) und (2.4)

$$\bar{p}_0 = \frac{1}{2}gH \quad \text{und} \quad \bar{p}' = \frac{q}{2}. \quad (2.11)$$

Da jedoch nicht der Druck selbst, sondern die Druckunterschiede den Fluidtransport beeinflussen, wird im Besonderen auf den gemittelten Druckgradienten geachtet. Teilchen in einer bestimmten Wassertiefe (gemessen ab η) erfahren denselben statischen Druck. Horizontale Veränderungen werden daher nur von der Veränderung der Oberflächengeometrie beeinflusst. In Formeln ausgedrückt, können die statischen Druckänderungen mit

$$\nabla p_0 = g\nabla \eta \quad \text{und} \quad \partial_z p_0 = -g \quad (2.12)$$

bestimmt werden. Diese Werte sind unabhängig von der Wassertiefe, so dass sie auch den vertikalen Durchschnittswert bilden.

Für den gemittelten nichthydrostatischen Druckgradienten können unter Annahme des linearen vertikalen Verlaufs folgende Gleichungen aufgestellt werden:

$$\overline{\nabla p'} = \frac{1}{H} \int_{-h}^{\eta} \nabla p' \, dz = \nabla \bar{q} \frac{H}{2} + \bar{q} \nabla \eta, \quad (2.13)$$

$$\overline{\partial_z p'} = \frac{1}{H} \int_{-h}^{\eta} \partial_z p' \, dz = -\bar{q}. \quad (2.14)$$

2. Physikalische Modellierung

Hierbei beschreibt $\bar{q} := q/H$ das Verhältnis des dynamischen Bodendrucks zur Gesamtwassertiefe¹. Die Herleitung ist in Anhang A.1 zu finden. In der horizontalen Druckveränderung spielen außer der Veränderung der Oberflächenauslenkung auch die vorherrschenden Bathymetriegradien und die Veränderung des Bodendrucks selbst eine Rolle. Durch die Annahme des linearen Druckverlaufs in z -Richtung herrscht unabhängig von der Tiefe dieselbe vertikale Veränderung und es gilt $\overline{\partial_z p'} = \partial_z p'$.

2.4. Erhaltungsgleichungen

Für die mathematische Beschreibung des Modells werden die Kontinuitätsgleichung und die Bewegungsgleichungen aus der Strömungsmechanik herangezogen. Sie werden hier in einer dem vertikal gemittelten Modell entsprechenden Schreibweise dargestellt.

2.4.1. Kontinuitätsgleichung

Die Kontinuitätsgleichung hält fest, dass weder Masse verschwindet, noch aus dem Nichts entsteht. Mit der Annahme der Inkompressibilität (2.1) ist die Massenerhaltung durch die Divergenzfreiheit der Geschwindigkeit gegeben:

$$\partial_x v_x + \partial_y v_y + \partial_z v_z = 0. \quad (2.15)$$

Über die Tiefe integriert kann diese Gleichung mit Hilfe der vertikal gemittelten Geschwindigkeitskomponenten und den kinematischen Randbedingungen in

$$\partial_t \eta + \nabla \cdot (\mathbf{u}H) = 0 \quad (2.16)$$

überführt werden. Die Herleitung ist in Anhang A.2 zu finden. Die horizontale Divergenz des Flusses $\mathbf{u}H$ steht also mit der zeitlichen Veränderung der Fluidtiefe, hier gegeben durch die Auslenkung des Wasserspiegels, im Gleichgewicht.

2.4.2. Bewegungsgleichungen

In den Bewegungsgleichungen werden zeitliche und räumliche Veränderungen des Systems unter Betrachtung aller äußeren Einwirkungen beschrieben. Sie dienen der Impulserhaltung und sind vom zweiten Newtonschen Gesetz abgeleitet, welches besagt, dass sich die Impulsänderung proportional zur einwirkenden Kraft verhält. Unter Annahme der Inkompressibilität (2.1) im vertikal gemittelten System lauten sie

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla) \mathbf{u} = \mathbf{F}_{xy} - \nabla p_0 - \overline{\nabla p'}, \quad (2.17)$$

$$\partial_t w + (\mathbf{u} \cdot \nabla) w = F_z - \partial_z p_0 - \partial_z p', \quad (2.18)$$

¹ Die Darstellung des gemittelten Druckgradienten in Abhängigkeit von $\bar{q} = q/H$ ist Kern der überarbeiteten Fassung, siehe Titelblattrückseite.

wobei $\mathbf{F}_{xy} = (F_x, F_y)$ und F_z spezifische Kraftkomponenten darstellen. \mathbf{F}_{xy} beschreibt äußere Einflüsse, wie die Corioliskraft, Reibung und Viskosität, während in F_z nur die Gravitation betrachtet wird:

$$\mathbf{F}_{xy} = -\mathbf{f} \times \mathbf{u} - \frac{gn^2 \mathbf{u} |\mathbf{u}|}{H^{4/3}} + \nabla \cdot (A_h \nabla \mathbf{u}), \quad (2.19)$$

$$F_z = -g. \quad (2.20)$$

Der Vektor \mathbf{f} beschreibt den Coriolisparameter in vertikaler Richtung, A_h den horizontalen Viskositätskoeffizienten und n parametrisiert die Bodenreibung nach Manning.

Das Einsetzen der Gleichungen (2.12) bis (2.14) führt zu

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla) \mathbf{u} = \mathbf{F}_{xy} - g \nabla \eta - \nabla \bar{q} \frac{H}{2} - \bar{q} \nabla \eta, \quad (2.21)$$

$$\partial_t w + (\mathbf{u} \cdot \nabla) w = \bar{q}. \quad (2.22)$$

Die statische Druckänderung in vertikaler Richtung wirkt entgegengesetzt zur Gravitation, so dass sich $\partial_z p_0$ und F_z gegenseitig aufheben.

Unter der hydrostatischen Annahme und der damit einhergehenden Vernachlässigung des hydrodynamischen Druckanteils, sind die Kontinuitätsgleichung (2.16) zusammen mit der Bewegungsgleichung (2.21) in ihrer reduzierten Form für $q \equiv \bar{q} \equiv 0$ als Flachwassergleichungen bekannt. Die vertikale Geschwindigkeitskomponente bleibt bei den Flachwassergleichungen unberücksichtigt.

3. Modelldiskretisierung

Zur Lösung der Gleichungen (2.16) und (2.21) - (2.22) wird hier ein Verfahren verwendet, das von [12] in einem Finite Differenzen (FD) Verfahren für ein dreidimensionales Modell basierend auf den Navier-Stokes-Gleichungen vorgestellt wurde. [62] befasste sich mit der effizienten Lösung der nichthydrostatischen Druckkomponente für diesen Ansatz für die Anwendung in einem mehrschichtigen FD Flachwassermodell. Ein derart über die Tiefe gemittelt FD Modell wird in [73] bei Studien verwendet, die sich mit dem Auflaufen der Wellen auf Land auseinandersetzen. Weitere Variante des nichthydrostatischen 2D-Modells basieren auf dem Finite Volumen (FV) Verfahren [15] oder der Finite Elemente (FE) Methode [72]. Diese FE Variante dient als Grundlage für diese Arbeit.

Durch die separate Berechnung des hydrostatischen und nichthydrostatischen Druckanteils können bestehende Ozeanmodelle, die auf den zweidimensionalen Flachwassergleichungen basieren, verwendet werden, um eine Zwischenlösung zu produzieren. Ein solches Modell wird nun um ein Modul erweitert, das in zusätzlichen Rechenschritten diese Zwischenergebnisse nach jedem Zeitschritt korrigiert. Somit wird die Berechnung in zwei Schritte aufgeteilt: die Lösung der Flachwassergleichungen und die nichthydrostatische Korrektur. Der erste Schritt wird von einem bestehenden Simulationsprogramm namens TsunAWI [3, 50] bewältigt. Diese Anwendung wurde am Alfred-Wegener-Institut, Helmholtz-Zentrum für Polar- und Meeresforschung (AWI) entwickelt, um für das deutsch-indonesische Tsunamifrühwarnsystem (GITEWS) Modellierungsaufgaben durchzuführen. In Abschnitt 3.1 werden einige grundlegende Fakten über die Diskretisierung und genutzte numerischen Methoden von TsunAWI dargestellt. In Abschnitt 3.2 wird anschließend die Erweiterung zur nichthydrostatischen Berechnung erläutert.

3.1. TsunAWI

Das Verfahren basiert auf einer in [27] beschriebenen $\mathcal{P}^1 - \mathcal{P}_{NC}^1$ FE Methode für unstrukturierte Gitter und arbeitet mit dem Leapfrog-Zeitschrittverfahren. In [3] ist eine Beschreibung der durchgeführten Veränderungen gegenüber [27] zu finden.

3.1.1. Räumliche Diskretisierung

Bei der FE Methode werden kontinuierliche Größen in einem Rechengebiet Ω durch eine endliche (finite) Anzahl von Stützwerten approximiert. Werte zwischen diesen Stützwerten werden mit Hilfe von Ansatzfunktionen interpoliert. Das Rechengebiet

3. Modelldiskretisierung

wird mit einer zulässigen, ebenen Triangulierung \mathcal{T} überdeckt, aufgespannt von der Knotenmenge $\mathcal{N} := \{n_1, \dots, n_N\}$ mit den Koordinaten $\mathbf{x}_i = \mathbf{x}(n_i)$. Die Triangulierung ist über die dreieckigen Elemente $\Omega^e \in \mathcal{T}$ mit

$$\Omega^e := \{\mathbf{x} \in \Omega : \mathbf{x} = \sum_{j=1}^3 \lambda_j \mathbf{x}(n_j^e), \lambda_j \geq 0, \sum_{j=1}^3 \lambda_j = 1, n_j^e \in \mathcal{N}\} \quad (3.1)$$

definiert. Jedes Element wird von drei Kanten begrenzt, deren Kantenmittelpunkte mit k_j^e bezeichnet werden. Die Kantenmittelpunkte aller Elemente $\Omega^e \in \mathcal{T}$ bilden die N_k -elementige Knotenmenge \mathcal{K} .

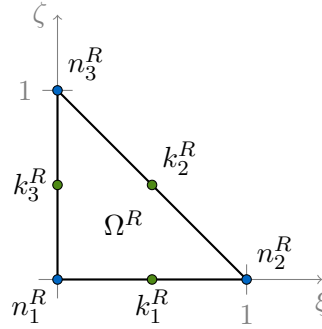


Abbildung 3.1.: Das Standardreferenzelement Ω^R mit den Eckknoten n_i^R und den Kantenmittelpunkten k_j^R , $1 \leq i, j \leq 3$.

In TsunAWI wird nach [27] und [35] ein Ansatz gewählt, bei denen die diskrete Oberflächenauslenkung $\eta_{\mathcal{T}}$ über die Knotenmenge \mathcal{N} und die Ansatzfunktionen $\varphi_i \in \mathcal{F}_{\mathcal{N}}$ definiert wird. Auf dem Standardreferenzelement in der ξ - ζ -Ebene, dargestellt in Abbildung 3.1, werden die Ansatzfunktionen mit

$$\varphi_1^R = 1 - \xi - \zeta, \quad \varphi_2^R = \xi, \quad \varphi_3^R = \zeta \quad (3.2)$$

beschrieben. Mit Hilfe von bijektiven Projektionsabbildungen $P_e : \Omega^R \rightarrow \Omega^e$ werden die Funktionen $\varphi_i \in \mathcal{F}_{\mathcal{N}}$ nun elementweise definiert, so dass die Eigenschaften

$$\varphi_i(\mathbf{x}(n_k)) = \varphi_i(\mathbf{x}_k) = \delta_{ik}, \quad \forall n_k \in \mathcal{N}, \quad (3.3)$$

und

$$\varphi_i|_{\Omega^e} \in \mathcal{P}^1(\Omega^e), \quad \forall \Omega^e \in \mathcal{T}, \quad (3.4)$$

erfüllt sind, wobei δ_{ik} das Kroneckerdelta und $\mathcal{P}^1(\Omega^e)$ den Raum der linearen Polynome auf Ω^e beschreibt. Mit (3.3) wird für die Stützstellen Konsistenz garantiert und es gilt $\eta(\mathbf{x}_i) = \eta_{\mathcal{T}}(\mathbf{x}_i)$. Die Kombination aus (3.3) und (3.4) führt dazu, dass sich der kompakte Träger $T_i := \text{supp}(\varphi_i)$ auf die kleine Anzahl von Elementen beschränkt, die den Knoten n_i enthalten.

Für die horizontalen Geschwindigkeit \mathbf{u} wird die Menge \mathcal{K} der Kantenmittelpunkte als Stützstellenmenge gewählt. Die entsprechenden Ansatzfunktionen $\psi_j \in \mathcal{F}_{\mathcal{K}}$ werden auf dem Standardreferenzelement mit

$$\psi_1^R = 1 - 2\zeta, \quad \psi_2^R = 2\xi + 2\zeta - 1, \quad \psi_3^R = 1 - 2\xi \quad (3.5)$$

beschrieben und mit den Projektionsabbildungen elementweise zusammengesetzt, so dass

$$\psi_j(\mathbf{x}(k_i)) = \delta_{ij}, \quad \forall k_i \in \mathcal{K}, \quad (3.6)$$

$$\psi_j|_{\Omega^e} \in \mathcal{P}^1(\Omega^e), \quad \forall \Omega^e \in \mathcal{T} \quad (3.7)$$

gilt. Der kompakte Träger $\text{supp}(\psi_j)$ beschränkt sich hierbei auf die zwei Elemente, deren Schnittmenge gerade der Kante mit Mittelpunkt k_j entspricht. In Abbildung 3.2 sind die Ansatzfunktionen beispielhaft dargestellt.

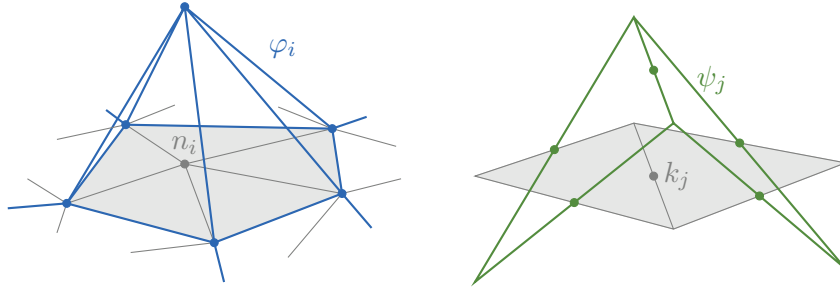


Abbildung 3.2.: Darstellung der Ansatzfunktionen φ_i und ψ_j mit grau eingefärbtem Träger.

Die Abbildungen der Approximationsfunktionen $\eta_{\mathcal{T}}$ und $\mathbf{u}_{\mathcal{T}}$ können nun als Linearkombinationen der Form

$$\eta_{\mathcal{T}}(\mathbf{x}) = \sum_{i=1}^N \eta_i \varphi_i(\mathbf{x}), \quad \mathbf{u}_{\mathcal{T}}(\mathbf{x}) = \sum_{j=1}^{N_k} \mathbf{u}_j \psi_j(\mathbf{x}) \quad (3.8)$$

dargestellt werden. Die Stützwerte werden in Vektoren $\eta \in \mathbb{R}^N$ und $\mathbf{u} \in \mathbb{R}^{N_k}$ zusammengefasst. Diese Vorgehensweise bezieht sich auch auf alle anderen Größen wie zum Beispiel der Wassertiefe H in der diskreten Darstellung.

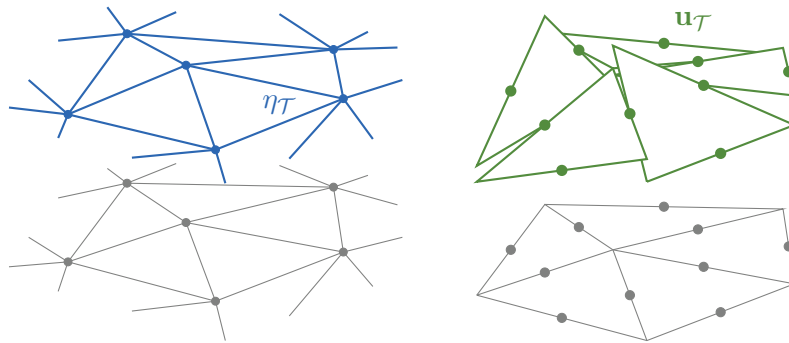


Abbildung 3.3.: Skizze der Approximationsfunktionen $\eta_{\mathcal{T}}$ und $\mathbf{u}_{\mathcal{T}}$.

In Abbildung 3.3 sind die Approximationsfunktionen $\eta_{\mathcal{T}}$ und $\mathbf{u}_{\mathcal{T}}$ skizziert. Die Oberflächenauslenkung $\eta_{\mathcal{T}}$ kann entlang der Elementkanten eindeutig bestimmt werden und

3. Modelldiskretisierung

ist über das gesamte Rechengebiet hinweg stetig: $\eta_{\mathcal{T}} \in C^0(\Omega)$. Bei der horizontalen Geschwindigkeit ist diese Stetigkeit nur innerhalb der Elemente und an den Kantenmittelpunkten garantiert, so dass ansonsten Sprünge auftreten können. Aufgrund dieser Sprünge werden die Ansatzfunktionen $\psi_j \in \mathcal{F}_{\mathcal{K}}$ als nichtkonforme $\mathcal{P}_{\text{NC}}^1$ -Funktionen bezeichnet.

Da sich die Tsunamibewegung in tiefem Gewässer von dem Verhalten der Welle beim Erreichen der Küste hinsichtlich Geschwindigkeit und Auslenkung stark unterscheidet, wird in Küstenregionen eine höhere Auflösung als auf hoher See benötigt. Um die Wellenpropagation sowohl im tiefen Gewässer als auch im Küstenbereich mit einem einzigen Modell berechnen zu können, wird in TsunAWI auf unstrukturierte Gitter zur Definition der Triangulierung zurückgegriffen. Unstrukturierte Gitter haben den Vorteil, dass keine einheitliche Gitterweite vorgegeben wird, wodurch der Übergang von grober zu feiner Auflösung nicht abrupt, sondern stufenlos stattfindet. Die Tatsache, dass keine gleichmäßige Struktur vorhanden ist, wirkt sich allerdings nachteilig auf die Optimierung in Bezug auf Zeiteffizienz aus. Die Algorithmen sind komplizierter, da zum Beispiel die Anzahl der Nachbarknoten und -elemente nicht einheitlich ist, was die Parallelisierung erschwert.

3.1.2. Zeitdiskretisierung

Für die Zeitdiskretisierung wird das Leapfrog-Verfahren mit äquidistanter Zeitschrittweite Δt verwendet. Dieses Verfahren besitzt eine Genauigkeit 2. Ordnung. Eine partielle Differentialgleichung der Form $\partial_t u + \partial_x f(u) = 0$ wird mit

$$\frac{u^{n+1} - u^{n-1}}{2\Delta t} + \partial_x f(u^n) = 0, \quad (3.9)$$

diskretisiert, wobei $u^n(x) = u(n\Delta t, x)$ beschreibt. Die Unbekannte u^{n+1} des nächsten Zeitschritts kann demnach explizit berechnet werden. Die Methode ist bedingt stabil: die Zeitschrittweite Δt wird abhängig von der Raumdiskretisierung durch das CFL-Kriterium [14] beschränkt. Beim Leapfrog-Verfahren kann eine künstliche Mode entstehen. Der hier verwendete Asselin-Robert-Filter [51, 5] dämpft selektiv die hohen Frequenzen und unterdrückt die künstliche Mode, wodurch jedoch die Genauigkeit des Verfahrens verringert wird [33].

3.1.3. Überflutung

Um eine unnatürliche Reflexion an der Küste zu vermeiden, wird in TsunAWI die Wellenausbreitung einschließlich Überflutung berechnet. Das Rechengebiet umfasst somit viele Landknoten mit Wassertiefe $H^0 = 0$. Durch Überflutung ist es möglich, dass trockene Knoten während der Simulation zu nassen Knoten mit $H > 0$ transformiert werden. Auch die Umkehrung ist möglich: nasse Knoten können unter Umständen trocken fallen, wenn das Wasser sich zurückzieht. Angeregt durch [39] werden für die Überflutung Modellgrößen von nassen Knoten auf direkt angrenzende Landknoten extrapoliert. In Abschnitt 5.3 wird die Thematik der trockenen und nassen Knoten noch einmal aufgegriffen.

3.1.4. Flachwassergleichungen

In seiner ursprünglichen Form löst TsunAWI nun mit Hilfe der oben genannten Methoden die Flachwassergleichungen, die für $\bar{q} \equiv 0$ mit den Gleichungen (2.16) und (2.21) übereinstimmen:

$$\frac{\eta^{n+1} - \eta^{n-1}}{2\Delta t} + \nabla \cdot (\mathbf{u}^n H^n) = 0, \quad (3.10)$$

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^{n-1}}{2\Delta t} + (\mathbf{u}^n \cdot \nabla) \mathbf{u}^n = -\mathbf{f} \times \mathbf{u}^n + \nabla \cdot (A_h \nabla \mathbf{u}^{n-1}) - \frac{gn^2 |\mathbf{u}^n| \mathbf{u}^{n+1}}{\sqrt[3]{(H^n)^4}} - g \nabla \eta^n. \quad (3.11)$$

3.1.5. Anfangs- und Randbedingungen

Für die Lösung der partiellen Differentialgleichungen müssen Anfangs- und Randbedingungen gesetzt werden. Der Ausgangsstatus für die Oberflächenauslenkung η und die horizontalen Geschwindigkeit \mathbf{u} wird mit

$$\begin{aligned} \eta(0, \mathbf{x}) &= \eta_0(\mathbf{x}), \\ \mathbf{u}(0, \mathbf{x}) &= \mathbf{u}_0(\mathbf{x}), \end{aligned} \quad \forall \mathbf{x} \in \Omega \quad (3.12)$$

vorgegeben. Um die Wellenbewegungen in einem Gebiet Ω zu berechnen, muss die horizontale Geschwindigkeit an den Gebietsrändern kontrolliert werden. Hierbei wird zwischen zwei Arten von Rändern unterschieden: Offene Ränder Γ_o , über die das Fluid das Gebiet verlassen kann und undurchlässige Ränder Γ_u . Es gilt $\partial\Omega = \Gamma = \Gamma_u \cup \Gamma_o$. Die Geschwindigkeiten werden sowohl am offenen Rand Γ_o , als auch an fluidundurchlässigen Grenzen Γ_u mit Neumann-Randbedingungen gesteuert. Kann die Flüssigkeit nicht durch einen Rand hindurch, so muss die Geschwindigkeit in Normalenrichtung für alle $t \in (0, t_{\text{end}}]$ verschwinden:

$$\mathbf{u}(t, \mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) = 0, \quad \forall \mathbf{x} \in \Gamma_u. \quad (3.13)$$

Der Vektor $\mathbf{n}(\mathbf{x})$ beschreibt dabei den nach außen zeigenden Normalenvektor orthogonal zum Rand Γ . Bei einem offenen Rand wird sichergestellt, dass das Fluid das Gebiet Ω verlassen kann ohne reflektiert zu werden. Dabei wird angenommen, dass die Ausbreitungsgeschwindigkeit c der Welle mit der für Flachwasserwellen $c_{\text{sw}}(t, \mathbf{x}) = \sqrt{gH(t, \mathbf{x})}$ übereinstimmt. Unter Berücksichtigung der Volumenerhaltung wird die Randbedingung für alle $t \in (0, t_{\text{end}}]$ mit

$$\mathbf{u}(t, \mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) = \frac{\eta(t, \mathbf{x})}{H(t, \mathbf{x})} c_{\text{sw}}(t, \mathbf{x}), \quad \forall \mathbf{x} \in \Gamma_o \quad (3.14)$$

beschrieben.

3.2. Korrektur der Geschwindigkeiten

Als Grundlage des Korrekturansatzes dienen nun die Oberflächenauslenkung η^{n+1} und die horizontale Geschwindigkeit $\tilde{\mathbf{u}}^{n+1}$, welche die Gleichungen (3.10) und (3.11) erfüllen. Tatsächlich entspricht $\tilde{\mathbf{u}}^{n+1}$ nur annähernd der hydrostatischen Lösung, da die Geschwindigkeitsvektoren vorhergehender Zeitschritte schon korrigierte Werte darstellen.

3.2.1. Zusätzliche Unbekannte

Die vertikale Geschwindigkeitskomponente w und der dynamische Bodendruckanteil q - im hydrostatischen Flachwasseransatz vernachlässigten Größen - werden nun bei den Korrekturtermen beachtet. Diese zusätzlichen Unbekannten werden in der FE Approximation an der Knotenmenge \mathcal{N} ausgewertet und dazwischen mit Hilfe der konformen Ansatzfunktionen aus $\mathcal{F}_{\mathcal{N}}$ interpoliert [72].

Bei der Beachtung von nichthydrostatischen Druckanteilen spielt die vertikale Geschwindigkeit eine wichtige Rolle. Um Kompatibilität zu gewährleisten, wird nun mittels Leapfrog-Verfahren die Geschwindigkeitskomponente \tilde{w}^{n+1} ermittelt, die analog zur Berechnung der horizontalen Geschwindigkeitskomponenten für $\bar{q} \equiv 0$ die vertikale Bewegungsgleichung (2.22) beschreibt:

$$\tilde{w}^{n+1} = w^{n-1} - 2\Delta t(\mathbf{u}^n \cdot \nabla)w^n. \quad (3.15)$$

Des Weiteren wird mittels der kinematischen Randbedingung (2.9) und $\delta_t h \equiv 0$ der Anteil am Grund bestimmt:

$$w_{-h}^{n+1} = -\mathbf{u}^n \cdot \nabla h^n. \quad (3.16)$$

Die horizontale Geschwindigkeit in dieser Gleichung ist im vorherigen Zeitschritt berechnet worden und somit im nichthydrostatischen Kontext zu verstehen. Bei der anschließenden Korrektur beschränkt sich die Modifikation auf den Oberflächenanteil w_η , der jedoch nicht explizit berechnet wird.

Mit $\tilde{\mathbf{u}}^{n+1}$ und \tilde{w}^{n+1} können die Geschwindigkeitskomponenten \mathbf{u}^{n+1} und w^{n+1} mit den Korrekturtermen

$$\mathbf{u}^{n+1} = \tilde{\mathbf{u}}^{n+1} - 2\Delta t \overline{\nabla p'} = \tilde{\mathbf{u}}^{n+1} - \Delta t (H^n \nabla \bar{q}^{n+1} + 2\bar{q}^{n+1} \nabla \eta^n), \quad (3.17)$$

$$w^{n+1} = \tilde{w}^{n+1} - 2\Delta t \partial_z p' = \tilde{w}^{n+1} + 2\Delta t \bar{q}^{n+1} \quad (3.18)$$

bestimmt werden, wenn das Verhältnis $\bar{q} = q/H$ bekannt ist. Die Herleitung für die Korrekturterme ist in Anhang A.3 zu finden.

Für die Berechnung von \bar{q} wird erneut die Kontinuitätsgleichung (2.15) herangezogen. Durch Multiplikation mit einer Testfunktion φ des Hilbertraums $H^1(\Omega)$ und Integration über das Volumen $V = \Omega \times H$ wird die Massenerhaltungsgleichung in eine schwache Formulierung übergeführt

$$\int_V (\partial_x v_x + \partial_y v_y + \partial_z v_z) \varphi \, dV = 0, \quad (3.19)$$

die per partieller Integration und unter Annahme (2.10) äquivalent ist zu

$$- \int_{\Omega} \mathbf{u} \cdot \nabla \varphi H d\Omega + \int_{\Omega} 2(w - w_{-h}) \varphi d\Omega + \int_{\Gamma_o} \mathbf{u} \cdot \mathbf{n} \varphi H d\Gamma_o = 0. \quad (3.20)$$

Das Randintegral aus Gleichung (3.20) wird nicht weiter betrachtet, da an offenen Rändern in Abschnitt 3.2.2 eine Dirichlet-Bedingung definiert wird. Durch Einsetzen der Korrekturterme (3.17) und (3.18) sowie einer Umsortierung der Terme folgt

$$\int_{\Omega} H^2 \nabla \bar{q} \cdot \nabla \varphi + 2H \bar{q} \nabla \eta \cdot \nabla \varphi + 4\bar{q} \varphi d\Omega = \frac{1}{\Delta t} \int_{\Omega} \tilde{\mathbf{u}} \cdot \nabla \varphi H - 2(\tilde{w} - w_{-h}) \varphi d\Omega. \quad (3.21)$$

In Anhang A.4 ist die Überführung der Gleichung (3.19) über (3.20) zu (3.21) ausführlich dargestellt. Da die Gesamtwassertiefe H an trockenen Knoten innerhalb des Rechengebiets verschwindet, weist Gleichung (3.21) eine degenierte elliptische Struktur auf. Mit Hilfe von entsprechenden Randbedingungen in Abschnitt 3.2.2 wird jedoch sichergestellt, dass beim Aufstellen der Gleichungen stets $H^2 > 0$ gilt.

Nun wird Gleichung (3.21) für die N konformen \mathcal{P}^1 -Funktionen $\varphi_i \in \mathcal{F}_{\mathcal{N}}$ aufgestellt, was zu einem linearen Gleichungssystem

$$\mathbf{A} \bar{\mathbf{q}} = \mathbf{b} \quad (3.22)$$

mit $\mathbf{A} \in \mathbb{R}^{N \times N}$ und $\bar{\mathbf{q}}, \mathbf{b} \in \mathbb{R}^N$ führt, das nach $\bar{\mathbf{q}}$ aufgelöst wird. Der i -te Eintrag von $\bar{\mathbf{q}}$ entspricht dann gerade dem Verhältnis q/H am Gitterknoten n_i . Ist dieses Verhältnis an den Stützstellen bekannt, so können die Geschwindigkeitskomponenten \mathbf{u} und w gemäß den Gleichungen (3.17) und (3.18) bestimmt werden. Bei einer hohen Anzahl an trockenen Knoten kann es von Vorteil sein, nur einen Teil des Gleichungssystems aufzustellen und zu lösen. Dieser Ansatz wird in Abschnitt 5.3.2 behandelt.

3.2.2. Anfangs- und Randbedingungen

Mit der horizontalen Initialgeschwindigkeit \mathbf{u}_0 aus (3.12) kann die vertikale Geschwindigkeit $w_0 = w(0, \mathbf{x})$ zum Zeitpunkt $t = 0$ mit Hilfe der Gleichungen (2.8) bis (2.10) ermittelt werden. Das Verhältnis q/H wird an offenen Rändern konsequent auf Null gesetzt. Außerdem werden die Geschwindigkeiten entlang dieser Randelemente nicht korrigiert, so dass eine Pufferzone eingehalten wird [72]. Zur Lösung des linearen Gleichungssystems müssen an den trockenen Landknoten, siehe Abschnitt 3.1.3, ebenfalls Randbedingungen gelten. Dabei wird dieselbe Kontrollstruktur wie an offenen Rändern verwendet. An undurchlässigen Rändern wird analog zur horizontalen Geschwindigkeit die Veränderung in Normalenrichtung untersagt. Es gilt also für alle $t \in (0, t_{\text{end}}]$:

$$\bar{q}(t, \mathbf{x}) = 0 \quad \forall \mathbf{x} \in \Omega_o \cup \Omega_t, \quad (3.23)$$

$$\nabla \bar{q}(t, \mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) = 0 \quad \forall \mathbf{x} \in \Gamma_u. \quad (3.24)$$

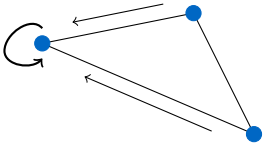
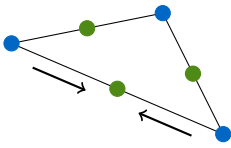
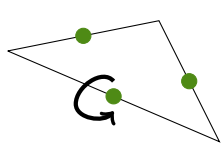
Bei dieser Beschreibung der Randbedingungen für \bar{q} beschreibt $\Omega_o = \{\Omega^e : \Omega^e \cap \Gamma_o \neq \emptyset\}$ und $\Omega_t = \{\Omega^e : \exists j \text{ mit } H_j^e = 0\}$.

3.3. Assemblierung der Matrizen

3.3.1. Massenmatrizen

Bei der Beschreibung der diskretisierten Größen mit Linearkombinationen wie in Gleichung (3.8) müssen im Zuge der FE Methode auf ein einzelnes Element $\Omega^e \in \mathcal{T}$ beschränkte Integrale der Ansatzfunktionen ausgewertet werden. Für Elemente außerhalb des entsprechenden Trägers verschwindet die Ansatzfunktion und somit das Integral. Andernfalls können die in Tabelle 3.1 aufgeführten lokalen Massenmatrizen aufgestellt werden. Der Flächeninhalt des Elements wird dabei mit $|\Omega^e|$ beschrieben. Während in der Massenmatrix \mathbf{M}^e der Ansatzfunktionen aus $\mathcal{F}_{\mathcal{N}}$ alle Werte der Elementknoten mit unterschiedlicher Gewichtung in die Berechnung eingehen, kommt bei der Diagonalmatrix \mathbf{L}^e die Orthogonalitätseigenschaft der Ansatzfunktionen $\mathcal{F}_{\mathcal{K}}$ zum Tragen. In der Matrix \mathbf{F}^e , welche die Interaktion von Ansatzfunktionen beider Funktionenmengen darstellt, werden pro Kante nur die Werte berücksichtigt, die an den Kantenendknoten gegeben sind.

Tabelle 3.1.: Auswertung der Integrale über Ω^e bezüglich der verschiedenen Kombinationen von Ansatzfunktionen mit $1 \leq i, j \leq 3, k = \text{mod}(j, 3) + 1$.

$\mathbf{M}^e \in \mathbb{R}^{3 \times 3}$	$\mathbf{F}^e \in \mathbb{R}^{3 \times 3}$	$\mathbf{L}^e \in \mathbb{R}^{3 \times 3}$
$\int_{\Omega^e} \varphi_i^e \varphi_j^e d\Omega^e = (\delta_{ij} + 1) \frac{ \Omega^e }{12}$	$\int_{\Omega^e} \psi_i^e \varphi_j^e d\Omega^e = (\delta_{ij} + \delta_{ik}) \frac{ \Omega^e }{6}$	$\int_{\Omega^e} \psi_i^e \psi_j^e d\Omega^e = \delta_{ij} \frac{ \Omega^e }{3}$
		

Zur Bestimmung der Werte bezüglich \mathcal{N} müssen lineare Gleichungssysteme gelöst werden, deren globale Massenmatrix \mathbf{M} aus einer Zusammensetzung der lokalen Matrizen \mathbf{M}^e besteht. Um diesem Rechenaufwand zu entgehen, wird die Berechnung approximiert, indem die Massenmatrix \mathbf{M}^e durch eine Diagonalmatrix approximiert wird, deren Einträge jenen von \mathbf{L}^e gleichen. Abbildung 3.4 stellt dieses Vorgehen graphisch dar. Auf diese Weise können η, w und w_{-h} explizit bestimmt werden, wie es bei der horizontalen Geschwindigkeit \mathbf{u} auch ohne Approximation der Fall ist.

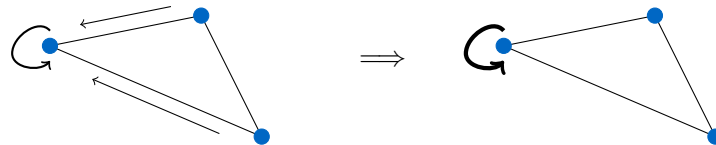


Abbildung 3.4.: Bei der Approximation der Massenmatrix werden die Gewichte aufsummiert und einem einzigen Stützwert übertragen.

3.3.2. Steifigkeitsmatrix und rechte Seite

Die Einträge der Matrix \mathbf{A} des Gleichungssystems (3.22) müssen in jedem Zeitschritt neu bestimmt werden. In der hier verwendeten knotenbezogenen Implementierung werden die Einträge der globalen Matrix zeilenweise gesetzt, um die Gleichungen für Knoten bezüglich der Randbedingung (3.23) besser sondieren zu können. Die Alternative der elementweisen FE Assemblierung lokaler Gleichungssysteme eignet sich für GPU-Implementierungen, jedoch ist das Aufstellen der globalen Matrix \mathbf{A} für MPI-parallele CPU-Implementierungen, wie sie hier angewendet werden, oft besser geeignet [40]. Der knotenbezogene Ansatz hat zudem den Vorteil, dass Löserroutinen und -bibliotheken einfach ausgetauscht und genutzt werden können, ohne dass beachtet werden muss, ob diese explizit für den FE Ansatz geschaffen wurden.

Die i -te Zeile des Gleichungssystems bezieht sich auf die Gleichung (3.21) mit der konformen Ansatzfunktion $\varphi_i \in \mathcal{F}_{\mathcal{N}}$ als Testfunktion. Da die FE Beschreibung des Bodendrucks ebenfalls auf der Stützstellenmenge \mathcal{N} in Kombination mit den konformen Ansatzfunktionen $\mathcal{F}_{\mathcal{N}}$ basiert, verschwinden alle Koeffizienten a_{ij} , deren Träger T_j keine Schnittmenge mit T_i aufweisen kann. Die Einträge der Steifigkeitsmatrix können mit

$$a_{ij} = \sum_{\Omega^e \subset T_i \cap T_j} \left(\sum_{k=1}^3 (H_k^e)^2 \nabla \varphi_i^e \cdot \nabla \varphi_j^e + 2H_j \nabla \eta^e \cdot \nabla \varphi_i^e + 4 \right) \frac{|\Omega^e|}{3}, \quad (3.25)$$

für $1 \leq i, j \leq N$ bestimmt werden, wobei die Kurzschreibweise der Gradienten der Form $\nabla \varphi_i^e = \nabla \varphi_i|_{\Omega^e}$ entspricht. Ebenso wird die Gesamtwassertiefe $H_k^e = H(n_k^e)$ lokal zugeordnet. Die Fläche des entsprechenden Elements Ω^e wird mit $|\Omega^e|$ angezeigt. Für die i -te Komponente der rechten Seite müssen nur diejenigen Geschwindigkeitskomponenten in Betracht gezogen werden, deren Ansatzfunktionen auf dem Träger T_i nicht vollständig verschwinden:

$$b_i = \frac{1}{\Delta t} \sum_{\Omega^e \subset T_i} \left(\sum_{j=1}^3 \sum_{k=1}^3 H_j^e \tilde{\mathbf{u}}_k^e \cdot \nabla \varphi_i^e (\delta_{kj} + \delta_{kl}) - 4(\tilde{w}_i - w_{-h,i}) \right) \frac{|\Omega^e|}{6}, \quad (3.26)$$

mit $l = \text{mod}(j,3)+1$, vergleiche \mathbf{F}^e aus Tabelle 3.1. Eine genauere Herleitung der Assemblierung von \mathbf{A} und \mathbf{b} ist in Anhang A.5 dargelegt.

Gilt für einen Knoten $n_i \in \Omega_o \cup \Omega_t$ die Randbedingung (3.23), so wird diese wie folgt umgesetzt: Die i -te Zeile der Matrix wird durch die entsprechende Zeile der Einheitsmatrix ersetzt und der Eintrag der rechten Seite auf Null gesetzt. Es gilt also:

$$a_{ij} = \delta_{ij} \quad \text{und} \quad b_i = 0. \quad (3.27)$$

Mit dieser Vorgehensweise bleibt das Randintegral aus Gleichung (3.20) unberührt. Die Randbedingung (3.23) gilt nicht nur für die trockenen Knoten, sondern auch für die nassen Nachbarknoten. Umgekehrt: Wenn einer der Nachbarknoten als trocken gilt, werden Matrixeinträge und rechte Seite mit Gleichung (3.27) berechnet. Wird ein Koeffizient a_{ij} mit Gleichung (3.25) bestimmt, so gilt demnach $H_k^e > 0$, für alle $\Omega^e \in T_i, k = 1, 3$.

4. Einfluss des nichthydrostatischen Druckterms

In TsunAWI wird aufgrund der hydrostatischen Annahme des Flachwassermodells der hydrodynamische Druckterm vernachlässigt. Das in Abschnitt 3.2 beschriebene Verfahren der nichthydrostatischen Erweiterung ist innerhalb TsunAWI-NH umgesetzt und erlaubt den Einfluss des hydrodynamischen Drucks zu berücksichtigen, während weiterhin mit über die Tiefe gemittelten Werten gerechnet wird. In diesem Kapitel wird der Einfluss des nichthydrostatischen Druckanteils anhand von Testbeispielen aufgezeigt, indem die Ergebnisse von TsunAWI und TsunAWI-NH mit Referenzwerten verglichen werden. Dabei werden im Folgenden die berechneten Größen von TsunAWI mit \cdot_{hyd} , die von TsunAWI-NH mit \cdot_{nh} gekennzeichnet. Die Ergebnisse der Beispiele 4.1 und 4.2 wurden schon in [4] veröffentlicht. Die numerischen Verfahren die in TsunAWI-NH genutzt werden, um das lineare Gleichungssystem (3.22) effizient zu lösen, werden dann in den anschließenden Kapiteln beschrieben.

4.1. Analytischer Testfall: Stehende Welle im Becken

Im ersten Beispiel wird eine stehende Welle im geschlossenen Becken berechnet. Die stehende Welle ist ein eindimensionales Problem, wird hier jedoch in einem 2D-Rechengebiet angeregt. Dabei wird angenommen, dass das bewegte Wasser sich wie ein ideales Fluid verhält. Dieser Testfall wurde auch in [62], [72] und [15] im Zusammenhang mit dem nichthydrostatischen Ansatz untersucht.

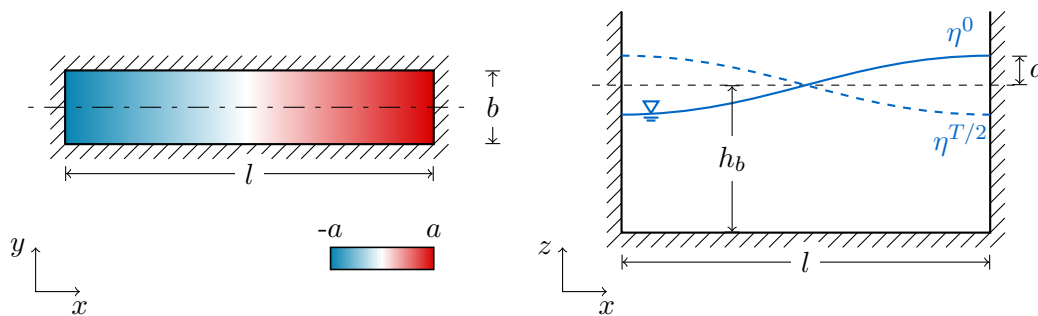


Abbildung 4.1.: Modellaufbau zum Testfall einer stehenden Welle im Becken. Links der Grundriss des Beckens, rechts die zugehörige Schnittansicht.

4. Einfluss des nichthydrostatischen Druckterms

4.1.1. Modellaufbau

Das Rechengebiet Ω umfasst ein rechteckiges Becken der Länge l und Breite b mit wasserundurchlässigen Beckenwänden $\delta\Omega = \Gamma_u$. An jeder Stützstelle wird eine konstante Referenztiefe h_b gesetzt, so dass der Bathymetriegradient ∇h des ebenen Beckenbodens verschwindet. Der Modellaufbau ist in Abbildung 4.1 dargestellt. Das Verhalten an den undurchlässigen Rändern wird mit den Neumannbedingungen aus (3.13) und (3.24) vorgeschrieben. Die Initialauslenkung einer stehenden Welle mit Wellenlänge $\lambda = 2l$ wird mit Hilfe einer Kosinusfunktion idealisiert:

$$\eta^0(x, y) = -a \cos\left(\frac{2\pi x}{\lambda}\right). \quad (4.1)$$

Die Anfangsgeschwindigkeit wird in allen Komponenten auf Null gesetzt. Mit dieser Initialisierung entstehen zwei Wellen mit gleicher Amplitude und Frequenz, die sich in entgegengesetzte Richtungen bewegen. Somit bildet sich eine stehende Welle mit einem Wellenknoten bei $x = l/2$ und Wellenbäuchen an den Beckenrändern, $x \in \{0, l\}$. Da das Wasser als ideales Fluid behandelt wird, wird die Bewegung allein durch den Druckgradienten angetrieben.

Tabelle 4.1.: Daten zum Modellaufbau der stehenden Welle im Becken.

	Parameter	Variable	Größe	Einheit
Abmessungen	Beckenlänge	l	10	m
	Beckenbreite	b	4	m
	Wellenlänge	λ	20	m
	Amplitude	a	0.1	m
Diskretisierung	Auflösung	Δx	~ 0.125	m
	Zeitschrittweite	Δt	0.01	s
	Simulationsdauer	T_{end}	20	s
Testläufe	minimale Beckentiefe	h_{min}	0.5	m
	maximale Beckentiefe	h_{max}	15.0	m
	Inkrement	Δh	0.25	m

4.1.2. Vergleich der Modellergebnisse

Am Beispiel der stehenden Welle kann durch Variation der Tiefe h_b gezeigt werden, dass der nichthydrostatische Druckanteil beim Anwachsen des Verhältnisses H/λ einen entscheidenden Einfluss gewinnt. Das Anwachsen von H/λ beschreibt den Übergang von einem Flachwasserbereich zu tieferem Gewässer. Beim Vergleich zwischen hydrostatischem und nichthydrostatischem Modell wird das Verhalten der Phasengeschwindigkeit c betrachtet, die mit

$$c = \frac{\lambda}{T} \quad (4.2)$$

berechnet wird, wobei T die Periodendauer der Wellenbewegung beschreibt. Als Referenz c_{ref} dient die nach Gleichung (2.5) analytisch bestimmte Phasengeschwindigkeit. Alle Rechendurchläufe werden mit fixen Werten l , b , a und λ durchgeführt. Zwischen den Läufen wird die Wassertiefe h_b variiert, beginnend mit der Tiefe h_{min} . Diese Wassertiefe wird dann iterativ um Δh vergrößert, bis der letzte Lauf mit $h_b = h_{\text{max}}$ durchgeführt wird. Die entsprechenden Werte sind der Tabelle 4.1 zu entnehmen. Die Veränderung der Wassertiefe zieht eine Veränderung des Verhältnisses H/λ nach sich, da die Wellenlänge über alle Läufe hinweg konstant ist. Anhand der jeweiligen Periodendauer T kann die Phasengeschwindigkeit nach (4.2) bestimmt werden. Alle Fälle werden sowohl mit TsunAWI, als auch mit TsunAWI-NH berechnet. Die Lösungen werden in Abbildung 4.2 mit der Ausbreitungsgeschwindigkeit c_{ref} aus Gleichung (2.5) beziehungsweise dem Grenzwert $c_{\text{sw}} = \sqrt{gH}$ verglichen.

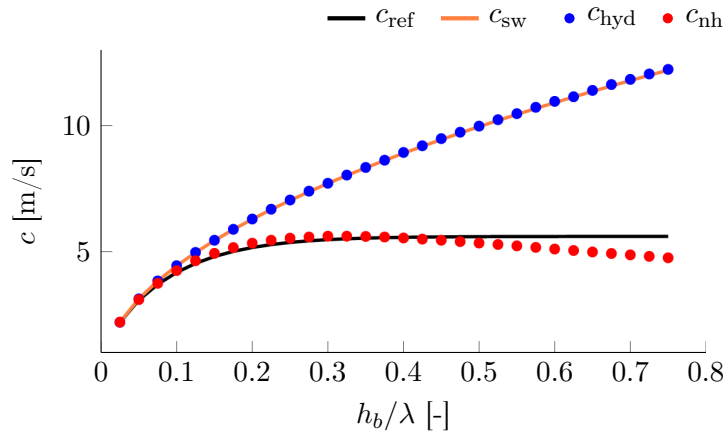


Abbildung 4.2.: Vergleich von hydrostatischen und nichthydrostatischen Modellergebnissen mit der analytischen Lösung von c als Referenz.

Es ist nicht überraschend, dass die aus dem Flachwassermodell resultierenden Werte für alle H/λ auf dem Graphen von c_{sw} liegen. Allerdings stimmt diese Lösung nur für $H/\lambda \ll 1$ mit der analytischen Lösung überein. Während die analytische Lösung bei anwachsender Wassertiefe stagniert, steigt beim Flachwassermodell die Fortpflanzungsgeschwindigkeit unaufhörlich an. Mit TsunAWI kann für $H/\lambda < 0.1$ ein gutes Ergebnis erwartet werden. Die Kurve basierend auf den Ergebnissen von TsunAWI-NH bietet trotz der Vereinfachung der über die Tiefe gemittelten Werten für $H/\lambda < 0.4$ eine gute Annäherung der analytischen Lösung, fällt danach jedoch ab. Verglichen mit dem Flachwassermodell liefert TsunAWI-NH für anwachsendes H/λ das bessere Ergebnis. In [62] wird gezeigt, dass eine mehrschichtige Betrachtungsweise noch bessere Ergebnisse liefert. Darauf wird hier jedoch nicht weiter eingegangen.

Abbildung 4.3 zeigt eine Zeitserie für das Experiment mit $h_b = 5$ m. Die nichthydrostatische Korrektur bremst das Flachwassermodell. Dennoch überschätzen beide Modelle die Ausbreitungsgeschwindigkeit c .

4. Einfluss des nichthydrostatischen Druckterms

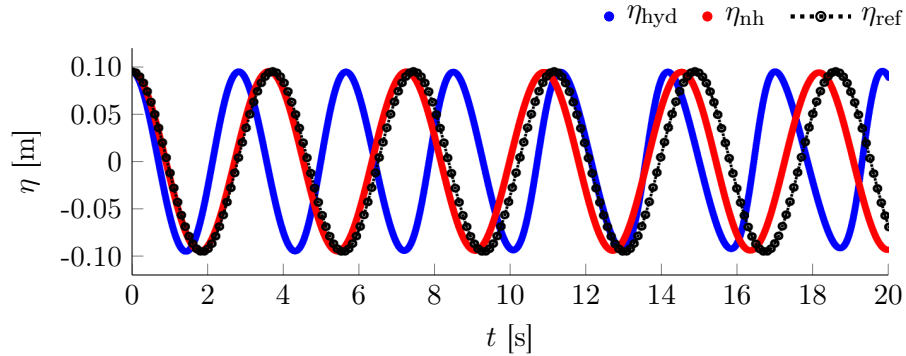


Abbildung 4.3.: Oberflächenauslenkung an der Position $(x_a, y_a) = (9.02, 1.47)$, aufgetragen über die Zeit bei einer Beckentiefe von $h_b = 5$ m.

4.2. Tankexperiment: Linearer Anstieg zur Küste

Im zweiten Beispiel wird die modellabhängige Beschreibung der Wellenbewegung beim Erreichen der Küste untersucht. Dabei bewegt sich eine einzelne Welle erst über einen Bereich mit konstanter Referenzwassertiefe, gefolgt von einem Abschnitt in dem der Grund linear ansteigt, wie in Abbildung 4.4 skizziert. Die zu Beginn trockenen Knoten können während eines Rechenlaufs überflutet werden. Zu diesem Testfall stehen als Referenz Messdaten zur Verfügung, die in einem Laborexperiment ermittelt wurden, welches im California Institute of Technology in Pasadena, Kalifornien durchgeführt wurde [63], [64]. Die Messdaten werden für die Validierung von Tsunamimodellen bereitgestellt [65].

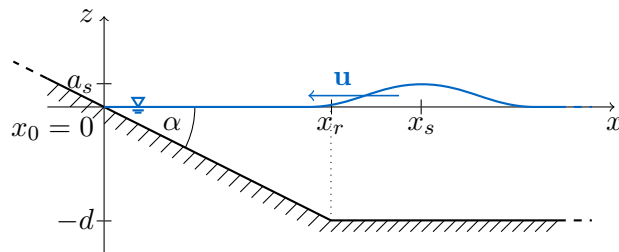


Abbildung 4.4.: Modellaufbau für das Experiment mit linearem Anstieg zur Küste.

4.2.1. Modellaufbau

In einem rechteckigen Becken herrscht eine konstante Wassertiefe d . An einem Ende bildet eine Rampe den linearen Anstieg um den Winkel α zur Küste hin nach. Auf der Rampe verschwindet an der Stelle $x = 0$ die Wassertiefe $h(0, y) = 0$ und trockenes Land beginnt. Nun wird ein einzelner Wellenberg mit der Amplitude a_s angeregt, welche sich auf die Küste zubewegt und diese anschließend überflutet. Alle räumlichen Größen in diesem Modellaufbau werden über die Tiefe d skaliert, so dass

sie dimensionslos sind. Mit $\tau = t\sqrt{g/d}$ wird sodann auch die Zeitachse angepasst. Der Einfachheit halber wird hier im Modell $d = 1$ m verwendet. Entsprechende Daten sind in Tabelle 4.2 aufgelistet.

Tabelle 4.2.: Modellaufbau zum Tankexperiment mit linearem Anstieg zur Küste.

	Parameter	Variable	Größe
Abmessungen	Tankausdehnung	x	[-10.0, 70.0]
	Tankausdehnung	y	[-0.5, 0.5]
	Tanktiefe	d	1
	Küstenlinie	x_0	0
	Rampenbeginn	x_r	19.85
	Anstieg der Rampe	α	$\arctan(d/x_r)$
Anregung	Amplitude 1	a_1	0.0185
	Amplitude 2	a_2	0.3
Diskretisierung	Auflösung	Δx	[0.1, 0.2]
	Zeitschrittweite	$\Delta \tau$	0.004
	Simulationsdauer	τ_{end}	31.3

Als Ausgangssituation wird wie in [68] die Oberflächenauslenkung eines einzelnen Wellenbergs mit der Gleichung

$$\eta^0(x, y) = a_s \operatorname{sech}^2 \left(\sqrt{\frac{3a_s}{4}} (x - x_s) \right) \quad (4.3)$$

modelliert, wobei die Maximalauslenkung a_s , unabhängig von y an der Position

$$x_s = \frac{d}{\tan(\alpha)} + \sqrt{\frac{4}{3a_s}} \operatorname{arccosh}(\sqrt{20}) \quad (4.4)$$

erfolgt. Um die Bewegung in Richtung Küstenlinie zu erzwingen, wird der Geschwindigkeitsvektor initial auf

$$\mathbf{u}^0(x, y) = \begin{pmatrix} -\eta^0 \sqrt{g/d} \\ 0 \end{pmatrix} \quad (4.5)$$

gesetzt. Beruhend auf dieser horizontalen Anfangsgeschwindigkeit können die kinematischen Randbedingungen (2.8) und (2.9) bestimmt und unter Verwendung der Annahme (2.10) kann die vertikale Geschwindigkeitskomponente initiiert werden. Während die Rampe überflutet werden kann, kontrollieren die Randbedingung (3.13) und (3.24) für Γ_u die undurchlässigen Beckenränder. Im Übergang zu den trockenen Knoten auf der Rampe wird nach (3.23) der nichthydrostatische Bodendruck q auf Null gesetzt. Mit Hilfe des Manningkoeffizienten $n = 0.01$ wird die glatte Oberfläche innerhalb des Tanks approximiert. Die Auflösung des unstrukturierten Gitters variiert, abhängig von der lokalen Referenztiefe h . Bei geringerer Wassertiefe ist eine höhere Auflösung erforderlich.

4. Einfluss des nichthydrostatischen Druckterms

4.2.2. Vergleich der Modellergebnisse

Es werden vier Rechenläufe gestartet: Mit TsunAWI und TsunAWI-NH wird jeweils das Experiment für die Amplituden a_1 und a_2 berechnet. Im Versuch mit $a_s = a_1$ liefern beide Ansätze sehr ähnliche Ergebnisse, die sehr gut mit den Beobachtungsdaten aus dem Laborexperiment übereinstimmen, siehe Abbildung 4.5.

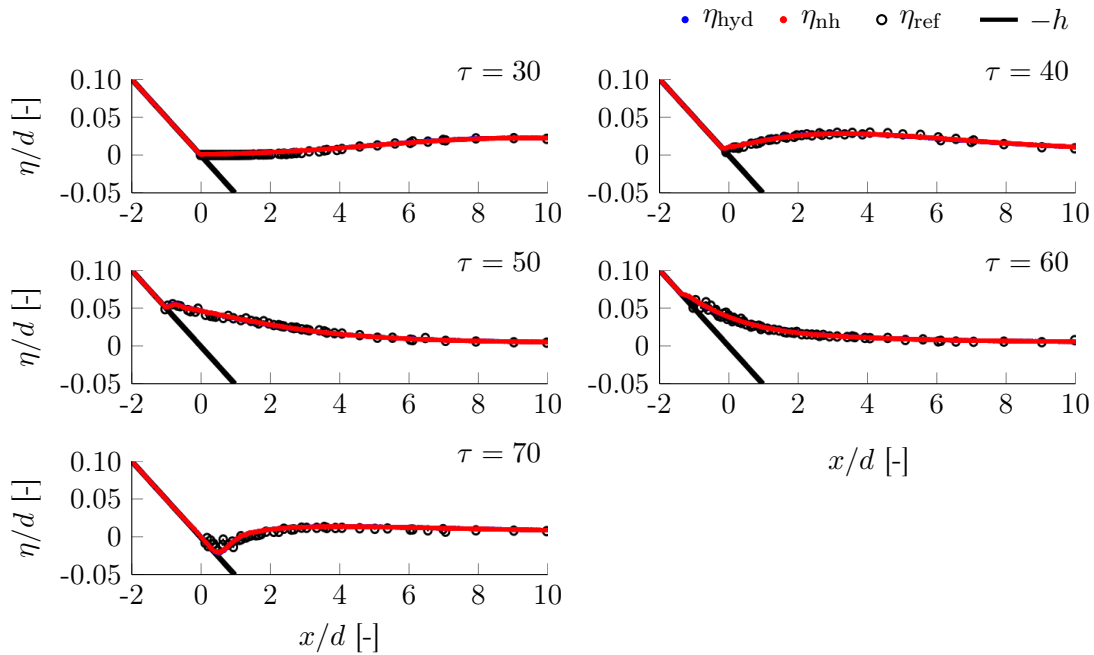


Abbildung 4.5.: Vergleich der Ergebnisse von TsunAWI und TsunAWI-NH im Beispiel $a_s=a_1$ mit Laborergebnissen als Referenz.

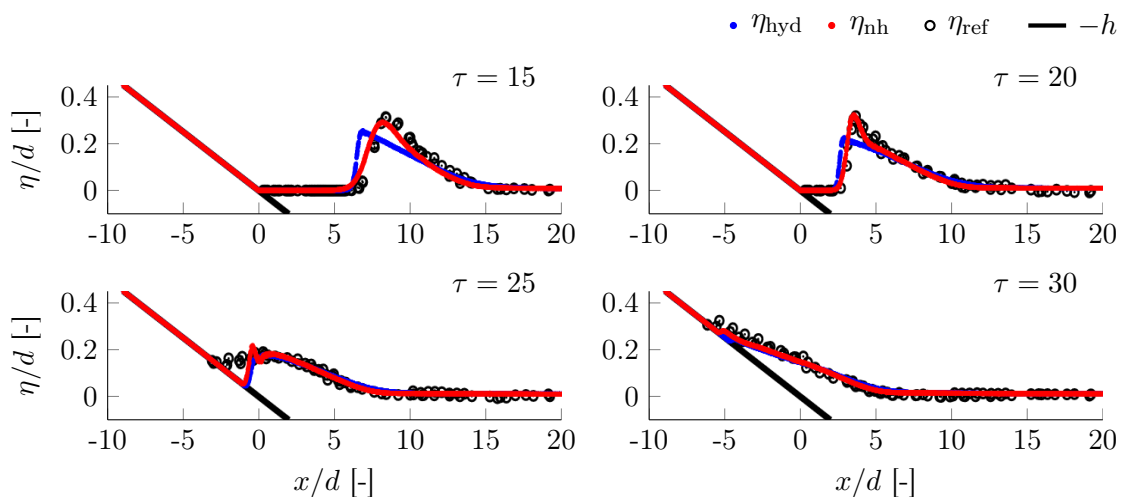


Abbildung 4.6.: Vergleich der Ergebnisse von TsunAWI und TsunAWI-NH im Beispiel $a_s=a_2$ mit Laborergebnissen als Referenz.

Interessanter ist der Fall $a_s = a_2$, welcher im Labor eine brechende Welle darstellte. Abbildung 4.6 zeigt im Querschnitt Momentaufnahmen zu vier verschiedenen Zeitpunkten. Offensichtlich liefert TsunAWI-NH eine bessere Annäherung an die Messdaten als die Lösung des hydrostatischen Modells. Aufgrund der kinematischen Randbedingungen (2.8) und (2.9) erfährt die Welle eine vertikale Bewegung. Da dieser Prozess mit dem Flachwassermodell nicht dargestellt werden kann, bildet die Welle eine unnatürlich steile Front aus, welche nur mit einer entsprechend kleinen Zeitschrittweite aufgelöst werden kann.

Die Abweichung der Modellergebnisse von den Referenzdaten wird mit Hilfe des empirischen Korrelationskoeffizienten

$$\delta_e(\mathbf{x}, \mathbf{y}) = \frac{1}{M-1} \sum_{i=1}^M \frac{(x_i - \bar{x})(y_i - \bar{y})}{\sigma(\mathbf{x})\sigma(\mathbf{y})} \quad (4.6)$$

und des Effektivwerts

$$E(\mathbf{x}, \mathbf{y}) = \sqrt{\frac{1}{M} \sum_{i=1}^M (x_i - y_i)^2} \quad (4.7)$$

für $\mathbf{x}, \mathbf{y} \in \mathbb{R}^M$ gemessen. Dabei beschreiben

$$\bar{x} = \frac{1}{M} \sum_{i=1}^M x_i \quad \text{und} \quad \sigma(\mathbf{x}) = \sqrt{\frac{1}{M-1} \sum_{i=1}^M (x_i - \bar{x})^2} \quad (4.8)$$

das arithmetische Mittel und die Standardabweichung. Die Dimension M entspricht der Anzahl der Messwerte. Die berechneten Zeitserien der Modellläufe werden zu den entsprechenden Messzeitpunkten interpoliert, so dass die Vektoren dieselbe Länge erhalten.

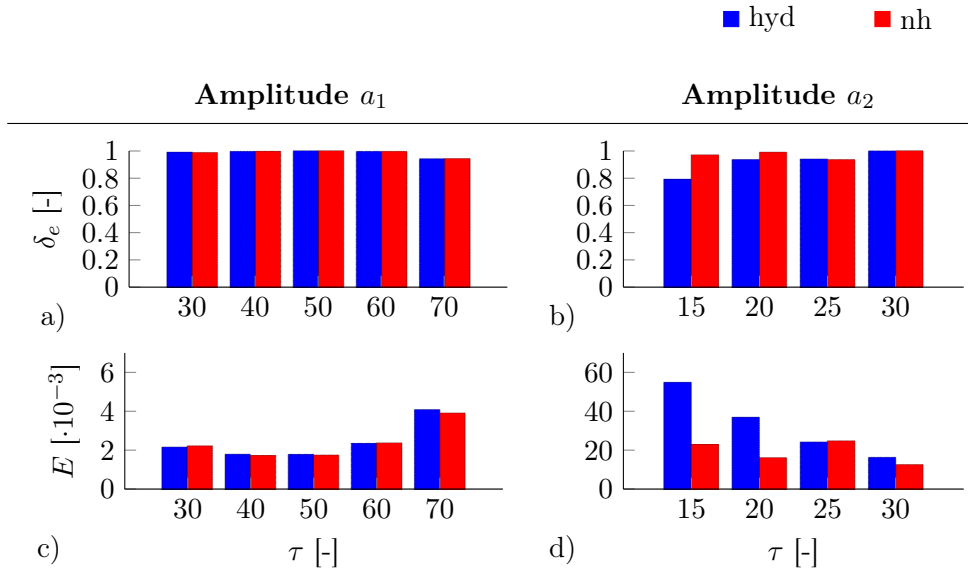


Abbildung 4.7.: Der empirische Korrelationskoeffizient δ_e und der Effektivwert E für die Anregungen mit den Amplituden a_1 und a_2 .

4. Einfluss des nichthydrostatischen Druckterms

Wie Abbildung 4.7 zeigt, erreicht TsunAWI im Rechenlauf mit der Amplitude a_1 sehr gute Ergebnisse, die der in TsunAWI-NH umgesetzte nichthydrostatische Ansatz kaum verbessert. Im Fall der brechenden Welle ist anfangs die Korrelation bezüglich des hydrostatisch modellierten Ergebnisses unter 80 Prozent und kann mit TsunAWI-NH auf 96 Prozent wesentlich erhöht werden. Zudem ist der Effektivwert bei den Ergebnissen von TsunAWI-NH in drei von vier Fällen deutlich geringer. In Überflutungssituationen, bei denen das hydrostatische Modell an seine Grenzen stößt, schafft es TsunAWI-NH trotz der ebenfalls vertikal gemittelten Werte die Wellenbewegung deutlich besser zu beschreiben.

4.3. Tankexperiment: Überströmung eines Hindernisses

In einem weiteren Beispiel wird untersucht, wie sich die modellierte Welle beim Überströmen eines Hindernisses verhält, das sich vollständig unter Wasser befindet. An der Technischen Universität in Delft wurde ein entsprechendes Laborexperiment durchgeführt, bei dem an verschiedenen Messstationen die Pegelstände aufgezeichnet wurden [10].

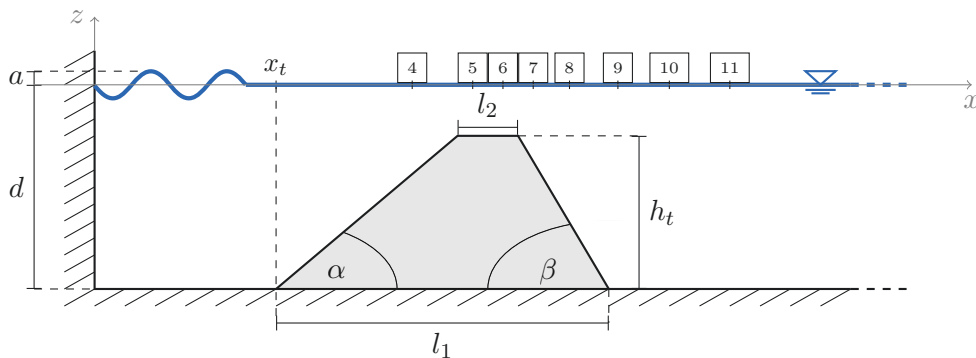


Abbildung 4.8.: Modellaufbau des Tankexperiments mit einem Unterwasserhindernis.

4.3.1. Modellaufbau

Am Rand eines Beckens der Wassertiefe d wird mit konstanter Frequenz $f = 1/T$ eine Welle mit der Amplitude a angeregt. Diese bewegt sich erst über einen Bereich mit ebener Wassertiefe, bevor sie ein trapezförmiges Hindernis überströmt. In Abbildung 4.8 ist eine Skizze des Modellaufbaus zu sehen. Das Hindernis hat dabei dieselbe Breite wie das Becken, so dass das Hindernis nicht horizontal umströmt werden kann. Die Beckenlänge ist so gewählt, dass innerhalb der Zeitintegration keine Reflexion am rechten Beckenrand bei $x = l$ stattfindet. Es wird die Anregung A aus [18], [62] untersucht. Die Abmessungen des Modellaufbaus, die Anregungsparameter sowie die hier verwendeten Diskretisierungsdaten sind in Tabelle 4.3 zu finden.

In der Ausgangssituation herrscht mit $\eta_0 \equiv 0$ und $\mathbf{u}_0 \equiv 0$ ein hydrostatisches Gleich-

4.3. Tankexperiment: Überströmung eines Hindernisses

gewicht. Die Wellenanregung am linken Beckenrand wird mit der Randbedingung

$$\eta(t, 0, y) = a \sin\left(\frac{2\pi t}{T}\right), \quad \forall (t, y) \in [0, T_{\text{end}}] \times \left[-\frac{b}{2}, \frac{b}{2}\right] \quad (4.9)$$

kontrolliert. Bei der nichthydrostatischen Modellierung wird zudem an diesem Rand die vertikale Geschwindigkeit auf

$$w(t, 0, y) = \frac{a\pi}{T} \cos\left(\frac{2\pi t}{T}\right), \quad \forall (t, y) \in [0, T_{\text{end}}] \times \left[-\frac{b}{2}, \frac{b}{2}\right] \quad (4.10)$$

und der nichthydrostatische Bodendruck $q(t, 0, y) = 0$ gesetzt. Für die horizontale Geschwindigkeit wird an den undurchlässigen Beckenrändern die Neumann-Bedingung für Γ_u aus Gleichung (3.13) verwendet. Der Rechenlauf wird sowohl mit TsunAWI, als auch TsunAWI-NH berechnet.

Tabelle 4.3.: Modellaufbau des Tankexperiments mit einem Unterwasserhindernis.

	Parameter	Variable	Größe	Einheit
Becken	Länge	l	150	m
	Breite	b	0.6	m
	Tiefe	d	0.4	m
Hindernis	Position	x_t	6.0	m
	Basis	l_1	11.0	m
	Kamm	l_2	2.0	m
	Höhe	h_t	0.3	m
	Anstieg	α	$\arctan(1/20)$	deg
	Abstieg	β	$\arctan(1/10)$	deg
Anregung	Amplitude	a	1.00	cm
	Periodendauer	T	2.02	s
Diskretisierung	Auflösung	Δx	[0.92, 2.67]	cm
	Zeitschrittweite	Δt	0.005	s
	Simulationsdauer	T_{end}	40.0	s

4.3.2. Vergleich der Modellergebnisse

Bis zum Erreichen des Hindernisses bewegt sich die harmonisch angeregte Welle mit der Phasengeschwindigkeit $c = \omega/k$. Dabei beschreibt $\omega = 2\pi/T$ die Kreisfrequenz und $k = 2\pi/\lambda$ die Wellenzahl. Beim Anströmen des Hindernisses wird die Welle stark abgebremst und die Welle nimmt die Form einer Sägezahnwelle an. Somit folgt die Welle keiner linearen Bewegung mehr, kann jedoch als Überlagerung harmonischer Wellen unterschiedlicher Länge λ_i beschrieben werden. Bei der anschließenden Beschleunigung nach der Überquerung des Hinderniskamms zerfällt die Welle, da die

4. Einfluss des nichthydrostatischen Druckterms

Phasengeschwindigkeit c und die Gruppengeschwindigkeit

$$c_g = \frac{\partial \omega}{\partial k} = c - \lambda \frac{\partial c}{\partial \lambda}, \quad (4.11)$$

mit der sich die Einhüllende fortplant, nicht mehr übereinstimmen.

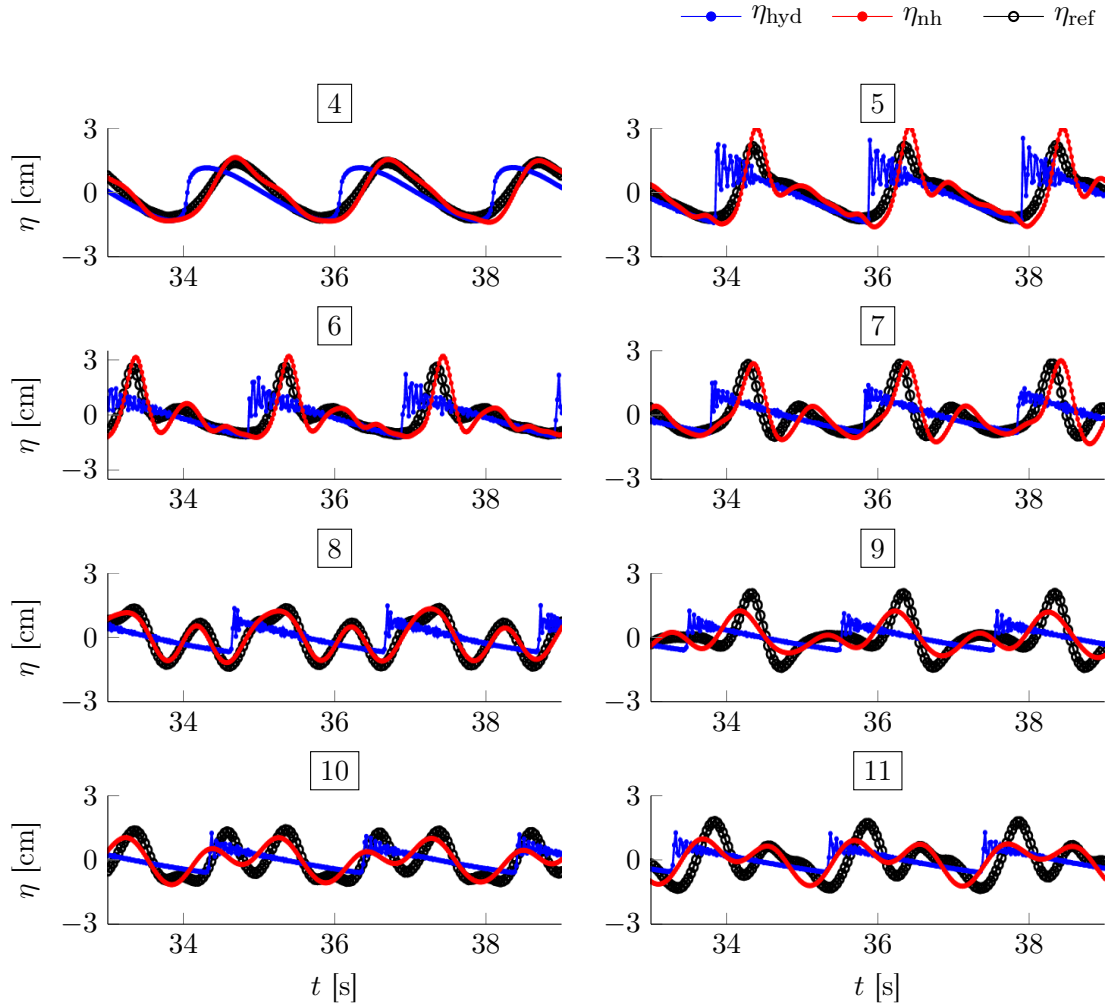


Abbildung 4.9.: Vergleich der Simulationsergebnisse von TsunAWI und TsunAWI-NH anhand von Zeitserien mit Laborergebnissen als Referenz.

Tabelle 4.4.: Position der Messstationen des Tankexperiments.

Messstation	i	4	5	6	7	8	9	10	11
Position	x_i [m]	10.5	12.5	13.5	14.5	15.7	17.3	19.9	21.0

Abbildung 4.9 zeigt die Ergebnisse der beiden Modellläufe als Zeitserien an acht Stationen mit Messwerten des Laborexperiments als Referenz. Die Position dieser Stationen sind in Tabelle 4.4 zu finden. Da im Flachwassermodell die Ausbreitungsgeschwindigkeit $c_{sw} = \sqrt{gH}$ nicht von der Wellenlänge λ abhängt, kann mit TsunAWI die Dispersion der Welle nicht dargestellt werden. Mit Gleichung (4.11) gilt über den ganzen Rechenlauf hinweg $c_g = c_{sw}$ und Phasen- und Gruppengeschwindigkeit stimmen stets überein. In allen acht Zeitserien wird die Sägezahnform beibehalten. TsunAWI-NH erzeugt deutlich bessere Ergebnisse. Das Zerfallen in zwei Wellenberge wird erfolgreich modelliert, in den Stationen hinter dem Hindernis wird jedoch eine Phasenverschiebung zu den Messergebnissen sichtbar, in der die Ausbreitung der Modellergebnisse schneller erfolgt.

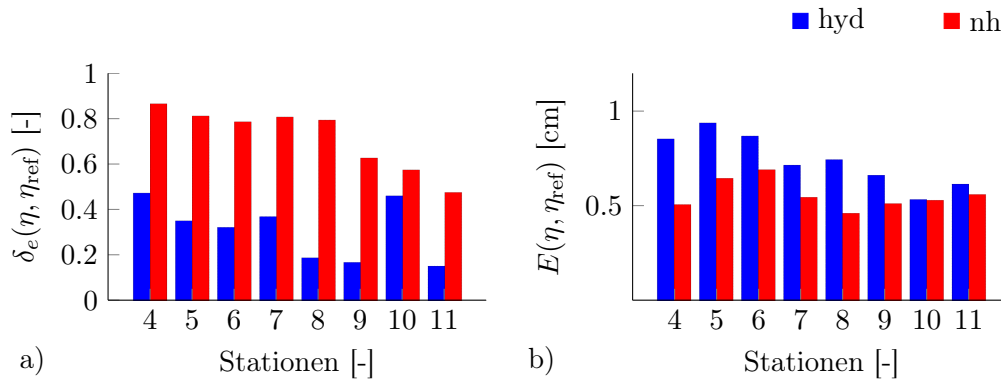


Abbildung 4.10.: a) Der Korrelationskoeffizient δ_e und b) der Effektivwert E für die Modellergebnisse in Relation zu den Messwerten.

In Abbildung 4.10 sind der empirische Korrelationskoeffizient $\delta_e(\eta, \eta_{ref})$ und der Effektivwert $E(\eta, \eta_{ref})$ für die acht Stationen dargestellt. Während der Korrelationskoeffizient bezüglich der nichthydrostatischen Modellergebnisse für die Stationen 4 bis 8 über 75 Prozent beträgt um dann langsam auf 48 Prozent abzufallen, wird mit dem Flachwassermodell kein einziges Mal die 50 %-Marke erreicht. Es überrascht nicht, dass der Effektivwert E bezogen auf die verschiedenen Messstationen für den nichthydrostatischen Modelllauf viel kleiner ausfällt als für den hydrostatischen. TsunAWI-NH zeigt also eine signifikante Verbesserung verglichen zu TsunAWI. In [62] wurde die Qualität der Modellergebnisse weiter verbessert, indem vertikal über zwei Schichten gerechnet wird. Auch bei der Verwendung eines Verfahrens höherer Ordnung können für dieses Beispiel bessere Ergebnisse erzielt werden [2].

5. Rechengitter und Matrixstruktur

Das Lösen des linearen Gleichungssystems (3.22) beansprucht einen Großteil der genutzten Rechenkapazitäten. Um den Rechenaufwand an dieser Stelle verringern zu können, hilft es den Zusammenhang zu beachten, der zwischen den Matrizen $\mathbf{A}(t)$ und dem Rechengitter besteht. Somit kann mit Veränderungen am Rechengitter eine Verbesserung der Lösungseigenschaften erzielt werden. Da sich zwar die Einträge $a_{ij}(t)$ der Matrix mit der Zeit verändern, aber gewisse Strukturen und Eigenschaften bestehen bleiben, wird der Einfachheit halber im Folgenden von der Matrix \mathbf{A} im Singular gesprochen.

5.1. Zusammenhänge

Die Struktur der Matrix hängt eng mit der Triangulierung \mathcal{T} des Rechengebiets zusammen. Diese kann einfach in den ungerichteten Graph $G_{\mathcal{T}}$ überführt werden, indem die Koordinaten der Gitterknoten vernachlässigt werden und nur noch der Einfluss der Knoten untereinander bestehen bleibt. Da der Träger der Ansatzfunktionen aus $\mathcal{F}_{\mathcal{N}}$ nur die Knoten anhängender Elemente umfasst, beschränkt sich der Einfluss eines Knotens auf die benachbarten Knoten und sich selbst. Unabhängig von der Geometrie zählt einzig der Zusammenhang der Knoten untereinander. In Abbildung 5.1 sind anhand eines einfachen Beispiels eine Triangulierung und der zugehörige Graph skizziert. Beim Graph wird dabei nur der Einfluss der Knoten $n_i \in V(G_{\mathcal{T}})$ untereinander betrachtet, ohne auf die Geometrie der Stützstellen einzugehen.

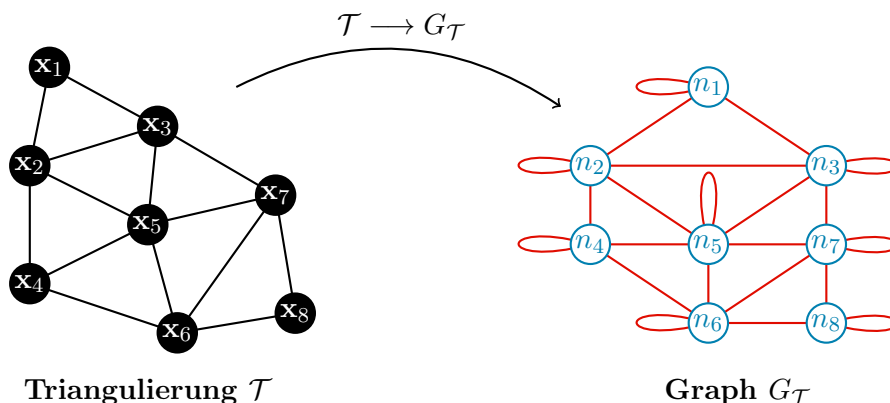


Abbildung 5.1.: Die Triangulierung \mathcal{T} wird in den Graph $G_{\mathcal{T}}$ überführt. Die Knoten $V(G_{\mathcal{T}})$ sind blau, die Verbindungen $E(G_{\mathcal{T}})$ rot dargestellt.

5. Rechengitter und Matrixstruktur

Mit den N Knoten aus $V(G_{\mathcal{T}})$ werden N Gleichungen zur Berechnung des dynamischen Bodendrucks q aufgestellt und die Matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ aus (3.22) assembliert. Die Zeile i der Matrix \mathbf{A} beschreibt die Gleichung für den hydrodynamischen Bodendruck q_i an der Stützstelle $n_i \in \mathcal{N}$. Nun tauchen in dieser Gleichung nur Stützwerte auf, deren zugehörige Knoten im Graph $G_{\mathcal{T}}$ mit dem Knoten n_i verbunden sind. Die Tatsache, dass jeder Knoten nur mit einem Bruchteil der Gesamtknotenanzahl direkt verbunden ist, wird darin widerspiegelt, dass fast alle Einträge a_{ij} , $i, j = 1, \dots, N$ der Matrix Nulleinträge sind. Es wird von einer dünnbesetzten Matrix gesprochen. Die Indexmenge S_A , welche die Position der Nichtnulleinträge definiert, kann mit

$$S_A = \{(i, j) : \exists \{n_i, n_j\} \in E(G_{\mathcal{T}})\} \quad (5.1)$$

beschrieben werden und es gilt

$$(i, j) \notin S_A \implies a_{ij} = 0. \quad (5.2)$$

Der Begriff Nichtnulleintrag bedeutet, dass der Eintrag einen Wert ungleich Null annehmen kann. Es ist jedoch durchaus möglich, dass der Wert dennoch verschwindet.

Existiert die Verbindung $\{n_i, n_j\} \in E(G_{\mathcal{T}})$, so gehören sowohl (i, j) , als auch (j, i) zu S_A . Da dies für alle bestehenden Verbindungen gilt, folgt eine struktursymmetrische Belegung der Matrix: Das Muster der Nichtnulleinträge ist symmetrisch, auch wenn die Werte bedingt durch Gleichung (3.25) die Symmetrieeigenschaft nicht erfüllen und $\mathbf{A} \neq \mathbf{A}^T$ gilt. Die Belegungsstruktur der Matrix hängt stark von der Gitterknotenindizierung im Rechengitter ab. Dieser Zusammenhang ist in Abbildung 5.2 an einem einfachen Beispiel dargestellt.

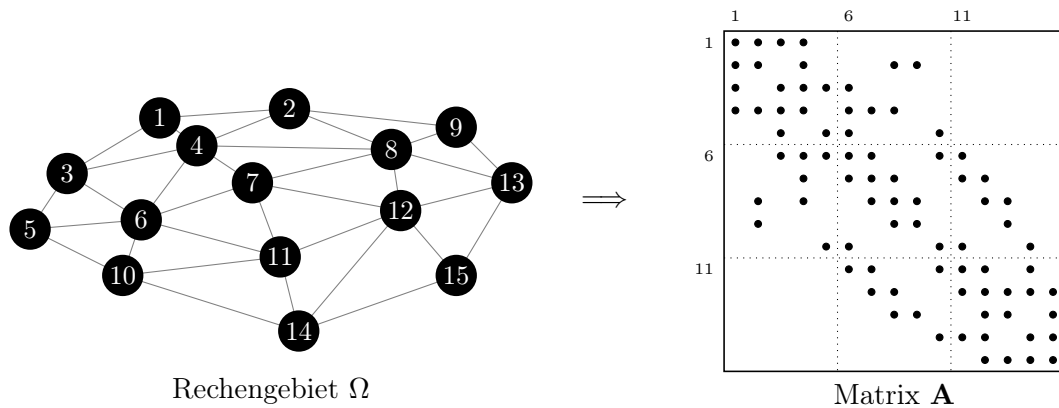


Abbildung 5.2.: Skizze von Rechengebiet und Matrixstruktur. Nichtnulleinträge werden durch einen Punkt gekennzeichnet.

Da es sich bei \mathbf{A} um eine dünnbesetzte Matrix handelt, lohnt es sich nicht sie als vollbesetzt zu betrachten, da die vielen Nulleinträge das Gleichungssystem in keinsten Weise beeinflussen, aber dennoch Speicherplatz verbrauchen. Um den Speicheraufwand zu verringern, wird die Matrix im CRS-Format (Compressed Row Storage) [8]

dargestellt, wie in Abbildung 5.3 entsprechend der Matrix aus Abbildung 5.2 skizziert. Statt einer $N \times N$ -Matrix mit N^2 Gleitkommazahlen doppelter Genauigkeit werden drei Vektoren angelegt: Die zwei ganzzahligen Vektoren \mathbf{r} und \mathbf{c} geben die Position eines Nichtnulleintrags an, während der Wert mit doppelter Genauigkeit im Vektor \mathbf{a} gespeichert wird. Der i -te Eintrag von \mathbf{r} beschreibt, an welcher Stelle die $(r_{i+1} - r_i)$ Spaltenindizes und die entsprechenden Werte der i -ten Zeile in den Vektoren \mathbf{c} und \mathbf{a} aufeinanderfolgend zu finden sind. Die Anzahl der Nichtnulleinträge entspricht gerade $M := |S_A| = r_{N+1} - 1$. Somit wird nur ein Speicherplatz von $4(N+1) + 4M + 8M < 8N^2$ Byte benötigt. Für die (sehr kleine) Matrix in Abbildung 5.2 bedeutet das mit $N = 15$ und $M = 81$ einen Speicheraufwand von 1036 Byte im CRS-Format statt der 1800 Byte, die für das Speichern der vollbesetzten Matrix benötigt wird.

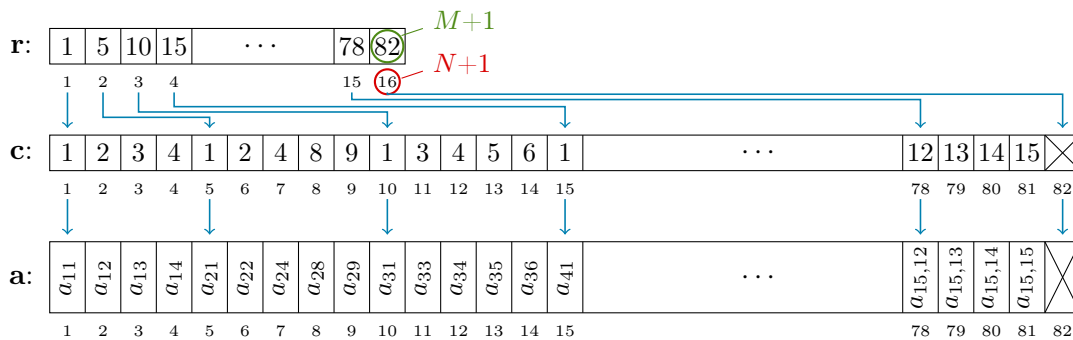


Abbildung 5.3.: Die Beschreibung der Matrix aus Abbildung 5.2 in CRS-Darstellung.

5.2. Umsortierung

Eine Umsortierung der Nummerierung von Gitterknoten und -elementen kann dabei helfen Datenlokalität zu erlangen. Da der Cache - ein schneller Zwischenspeicher der den direkten Zugriff auf den langsamen Hauptspeicher vermeidet - ganze Datenblöcke aus dem Hauptspeicher lädt, ist es sehr vorteilhaft, wenn verschiedene Daten, die häufig gemeinsam in Gleichungen auftauchen, nahe beieinander gespeichert werden. Somit werden viele relevanten Daten gemeinsam in den Cache geladen und die Anzahl der Kopiervorgänge aus dem Hauptspeicher wird verringert. Zudem führt eine Veränderung der Knotenindizierung zu einer Permutation der Matrix, wodurch sich die Indexmenge und die Struktur verändern. Dies kann bei der LU-Faktorisierung der Matrix mittels Gauß-Elimination hilfreich sein, da die Anzahl der auftretenden zusätzlichen Nichtnulleinträge davon abhängt, in welcher Reihenfolge die Eliminierung stattfindet [23]. In dieser Arbeit wird keine vollständige LU-Zerlegung durchgeführt. Doch im Falle einer unvollständigen Faktorisierung, siehe Abschnitt 8.1.2, bedeutet eine geringere Anzahl an zusätzlichen Nichtnulleinträgen, dass gegebenenfalls weniger Einträge unterdrückt werden. Abbildung 5.4 illustriert anhand der Belegungsstruktur das Auftreten neuer Nichtnulleinträge bei der Faktorisierung.

5. Rechengitter und Matrixstruktur

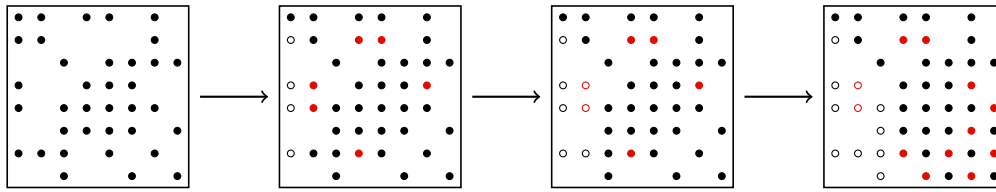


Abbildung 5.4.: Veränderung der Matrixstruktur bei der Gauß-Eliminierung. Eliminierte Einträge sind ohne Füllung, neu entstehende Nichtnull-Einträge sind rot gekennzeichnet.

Um den Zusammenhang zur Knotennummerierung zu veranschaulichen, wird nicht das Belegungsmuster sondern der zugehörige Graph betrachtet, wobei die Schleifen, die einen Knoten mit sich selbst verbinden, vernachlässigt werden. Wird ein Knoten während der Faktorisierung eliminiert, so wird eine Verbindung zwischen den benachbarten Knoten hergestellt, falls diese noch nicht existiert. Jede neue Verbindung entspricht direkt zwei zusätzlichen Einträgen. Indirekt können bei der weiteren Faktorisierung noch weitere Verbindungen und somit Nichtnull-Einträge folgen. Abbildung 5.5 zeigt die Folgen, abhängig davon, welcher Knoten eliminiert wird. Die Zahlen in den Knoten zeigen deren Grad $d_i = d(n_i)$, der die Anzahl der Nachbarknoten angibt. In Variante 1 wird ein Knoten mit minimalem Grad eliminiert, wobei keine neue Verbindung auftritt. Bei der Eliminierung von Knoten höheren Grades werden hier neue Verbindungen aufgebaut. In Variante 3 so viele, dass der neue Grad einiger benachbarten Knoten trotz Eliminierung sogar anwächst.

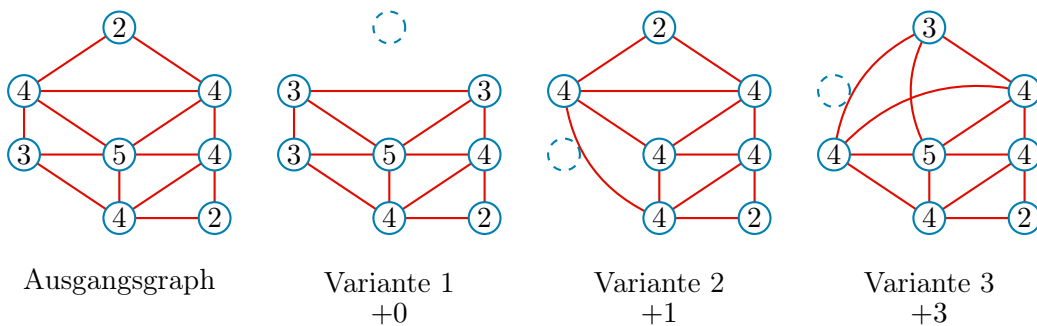


Abbildung 5.5.: Veränderung des Graphen bei der Eliminierung eines Eintrags.

Zwei der hier verwendeten Sortieralgorithmen orientieren sich in ihrer Herangehensweise an der Betrachtung des Grades. Eine Sortierung ordnet iterativ spaltenweise nach einer Approximation des minimalen Grades (COLAMD) mit Hilfe einer symbolischen LU-Zerlegung, welche nur mit dem Belegungsmuster, also ohne Werte arbeitet [17]. Der Reverse Cuthill-McKee Algorithmus (RCM) nummeriert ausgehend von einem beliebigen Startknoten n_1 dessen direkte Nachbarknoten mit aufsteigendem Grad durch und wiederholt dieses Verfahren mit den noch unnummerierten Nachbarknoten des Knotens n_2 und so fort. Nachdem alle Knoten einen Index zugewiesen bekommen haben, wird die so entstandene Nummerierung umgekehrt [23].

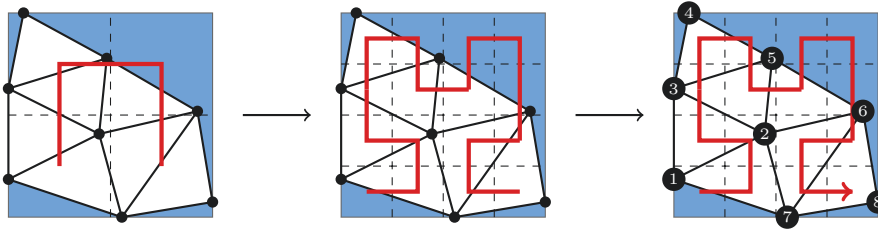


Abbildung 5.6.: Iterativ wird die Ordnung der Hilbertkurve erhöht, bis in jedem Segment höchstens ein Gitterknoten liegt. Die Nummerierung erfolgt in der Reihenfolge in der die Kurve die Segmente durchläuft.

Im Gegensatz dazu verfolgt eine weitere Methode einen geometrischen Ansatz, wobei die Knotennummerierung durch eine raumfüllende Kurve (SFC) definiert wird. Der Einsatz von raumfüllenden Kurven beim wissenschaftlichen Rechnen ist in [7] ausführlich beschrieben. Dabei wird nicht der Graph, sondern direkt das Diskretisierungsgitter betrachtet, da die Koordinaten der einzelnen Gitterknoten wichtig sind. Mit Hilfe einer Hilbertkurve wird das Rechengebiet in mehreren Schritten so lange unterteilt, bis in jedem der entstandenen Segmente höchstens ein Gitterknoten liegt, wie in Abbildung 5.6 skizziert. Die anschließende Nummerierung erfolgt in der Reihenfolge, wie die Segmente von der Kurve durchlaufen werden. Das SFC-Verfahren geht zwar nicht auf den Grad der Knoten ein, jedoch zeigt das Füllmuster der Matrix, dass die Nichtnulleinträge weitgehend nahe der Diagonalen angesiedelt sind.

Abbildung 5.7 zeigt die entsprechenden Belegungsstrukturen einer Matrix nach der Permutation basierend auf den verschiedenen Sortieralgorithmen.

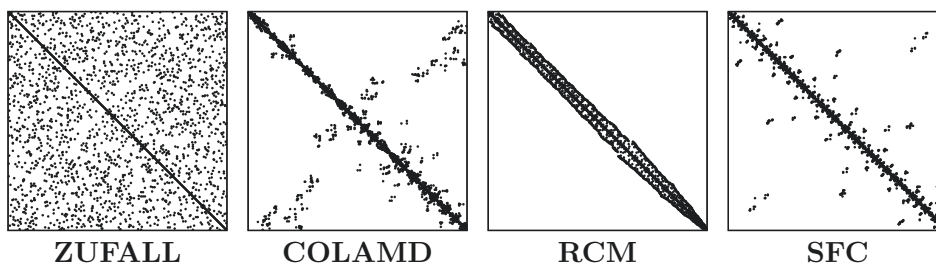


Abbildung 5.7.: Charakteristische Füllmuster der verschiedenen Nummerierungen.

Für das operationell genutzte Flachwassermodell TsunAWI wird standardmäßig die Knotennummerierung des Rechengitters mit Hilfe einer raumfüllenden Hilbert-Kurve umsortiert [50]. Der SFC-Ansatz hat sich im Vergleich zu Standardsortierungen in Bezug auf Datenlokalität bewährt [49]. In Abschnitt 9.2.4 wird der Einfluss verschiedener Sortierungen auf die Effizienz der unvollständige LU-Zerlegung untersucht.

5.3. Beschränkung des Rechengebiets

Da TsunAWI sich nicht auf die Propagationsphase beschränkt, sondern durchaus auch Überflutung modelliert, wird tatsächlich auf einem variablen Gebiet gerechnet.

Bei einer Klassifizierung der Knoten aus \mathcal{N} wird zwischen zwei Arten unterschieden: Ein Knoten n_i gilt als nass, falls

$$\eta_i(t) + h_i > \varepsilon_w \quad \text{mit} \quad \varepsilon_w > 0, \quad (5.3)$$

andernfalls handelt es sich um einen trockenen Knoten oder Landknoten. Da sich die Oberflächenauslenkung im Verlauf der Anwendung ändert, kann aus einem nassen Knoten ein trockener werden und umgekehrt, wie es bei einer Überflutung der Fall ist. Die Menge der nassen Knoten

$$\mathcal{N}_w(t) := \{n_i : \eta_i(t) + h_i > \varepsilon_w\}, \quad (5.4)$$

variiert deshalb mit der Zeit. Bei der Tsunamisimulation einschließlich Überflutung gibt es immer Stützstellen im Rechengitter, welche zu den trockenen Knoten zählen. An diesen trockenen Knoten sind die Stützwerte der Oberflächenauslenkung, der Geschwindigkeitskomponenten und des dynamischen Bodendrucks allesamt Null. Ein Gitterknoten kann nur dann von trocken zu nass übergehen, falls ihn mindestens eine Kante mit einem nassen Knoten verbindet. Somit finden ausschließlich in einem Teilgebiet $\Omega_v(t) \subseteq \Omega$ Veränderungen statt. Dieser Sachverhalt ist in Abbildung 5.8 dargestellt. Die zugehörige Knotenmenge $\mathcal{N}_v(t)$ enthält alle Stützpunkte, für die gilt:

$$\mathcal{N}_v(t) = \{\exists(i, j) \in \mathcal{E} : n_i \in \mathcal{N}_w(t) \vee n_j \in \mathcal{N}_w(t)\}. \quad (5.5)$$

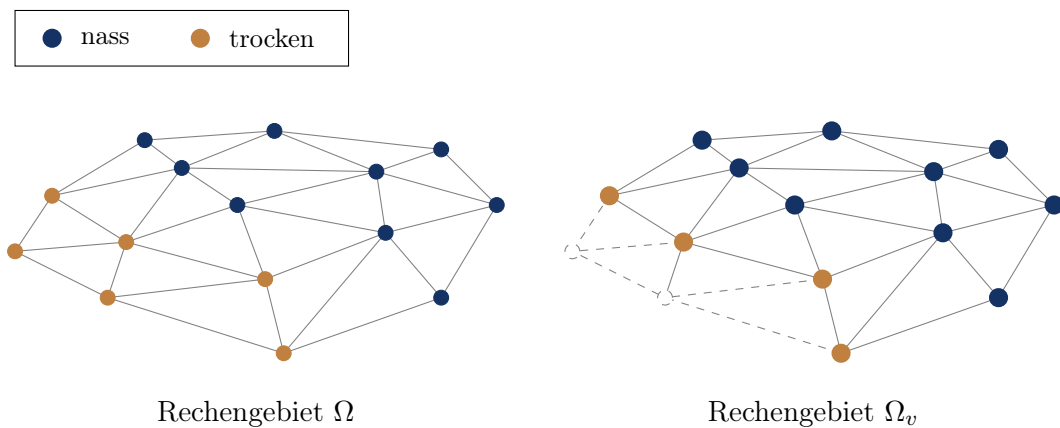


Abbildung 5.8.: Das variable Rechengebiet Ω_v ist ein Teilgebiet des Gesamtgebiets Ω .

5.3.1. Gesamtmatrix mit konstanten Strukturen

Da die Matrixeinträge im CRS-Format gespeichert werden, muss für die Matrix die Belegungsstruktur bestimmt werden, die besagt, wieviele und welche Spalten pro Zeile einen Nichtnulleintrag besitzen. Die Struktur der Matrix und die zugehörige Indexmenge S_A sind dabei eng an das Rechengitter samt Indizierung geknüpft, wie in Abschnitt 5.1 erläutert wurde. Da sich das Rechengitter während eines Programmablaufs nicht ändert, ist die Indexmenge schon in der Aufbauphase bekannt und die Matrix wird insoweit vorbereitet, dass Speicherbedarf und Zugriffsstrukturen angelegt werden. In der Berechnungsphase werden dann in jedem Zeitschritt die aktuellen Matrixeinträge an den entsprechenden Positionen neu gesetzt. Immer dieselben angelegten Felder für die Matrixstrukturen zu verwenden, führt dazu, dass diese bei Bedarf aus dem Hauptspeicher geholt werden müssen. Dies kann im Einzelfall länger dauern, als mit frisch angelegten Feldern zu arbeiten, welche sofort im Cache vorliegen.

Im Gleichungssystem werden mit Gleichung (3.23) trockene Knoten als Randknoten behandelt. Dementsprechend gilt für die Matrixeinträge und die rechte Seite Gleichung (3.27). Abbildung 5.9 zeigt das Füllmuster der Matrix aus Abbildung 5.8.

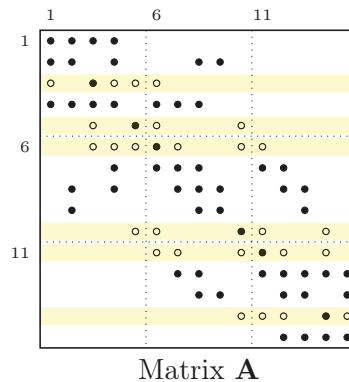


Abbildung 5.9.: Die Belegungsstrukturen der Matrix **A**. Die Zeilen der betrachteten trockenen Knoten sind gelb unterlegt. Kreise ohne Füllung stellen gespeicherte Nulleinträge der dünnbesetzten Matrix dar.

Um in jedem Zeitschritt ein Gleichungssystem mit konstanter Belegungsstruktur berechnen zu können, müssen diese Zeilen jedoch mitgeführt werden, da die entsprechenden Knoten im Verlauf der Rechnung überflutet werden können und in diesem Fall durchaus dynamischer Bodendruck entsteht. Insgesamt wird also ein größeres Gleichungssystem gelöst als eigentlich notwendig. Der Anteil des Teilgebiets Ω_v innerhalb des Gesamtgebiets wird im Folgenden anhand der aufspannenden Knotenmengen gemessen und mit

$$\kappa(t) := \frac{|\mathcal{N}_v(t)|}{|\mathcal{N}|}, \quad (5.6)$$

bezeichnet, wobei $|\cdot|$ die Mächtigkeit der jeweiligen Menge beschreibt. Da in Küstengegenden die Auflösung des unstrukturierten Gitters besonders hoch ist, gilt ein sehr großer Anteil der Knoten als trocken und $\kappa(t) \in [0, 1]$ ist entsprechend klein.

5.3.2. Teilmatrix mit variablen Strukturen

Um die Größe des Gleichungssystems auf die Problemgröße abzustimmen, gibt es die Möglichkeit, in jedem Zeitschritt ein neues System aufzusetzen. Die Struktur der reduzierten Matrix \mathbf{A}_v wird in diesem Fall immer wieder neu angelegt, da sich Dimension und Füllmuster ändern. Dies erfordert für gewöhnlich mehr Zeit, als nur die Einträge an die entsprechende Stelle zu speichern. Dafür können diese Werte direkt in Felder geschrieben werden, welche sich im Cache befinden.

Um eine einfache Assemblierung der Matrixstruktur zu ermöglichen, wird ausschließlich die Knotenmenge $\mathcal{N}_v(t)$ betrachtet, die das variable Rechengebiet Ω_v aufspannt. Zu dieser Knotenmenge gehören auch die trockenen Knoten, die den Abschluss des Teilgebiets bilden. Alle Nachbarknoten eines nassen Knotens sind somit im Teilgebiet enthalten und es muss keine Auswahl getroffen werden, welche Knoten betrachtet werden und welche nicht. Für die trockenen Knoten gilt die Randbedingung (3.23). In der variablen Besetzungsstruktur wird für die entsprechende Zeile nur der Diagonaleintrag belegt, die Nullwerte werden nicht gespeichert. Somit muss auch hier keine Auswahl getroffen werden. Diese Vereinfachung kann vorgenommen werden, da die verwendeten Lösungs- und Präkonditionierungsverfahren aufgrund der Unsymmetrie von \mathbf{A} keine symmetrische Struktur voraussetzen.

Die Knoten aus \mathcal{N}_v müssen neu nummeriert werden, damit keine Lücken entstehen. Wird ein Gitterknoten nicht beachtet, da er nicht in \mathcal{N}_v enthalten ist, so findet an dieser Stelle eine Indexverschiebung statt. Abbildung 6.9 zeigt die entsprechende Struktur der reduzierten Matrix \mathbf{A}_v . Um innerhalb der verteilten Matrix die Einträge den richtigen Partitionen zuzuordnen, wird die Struktur der Gesamtmatrix über die gesamte Integrationsdauer gespeichert, da sie zur Orientierung dient. Innerhalb des Lösungsverfahrens müssen jedoch im Vergleich zur Gesamtmatrix weniger Einträge gespeichert werden, so dass sich der gesamte Speicherbedarf in Grenzen hält.

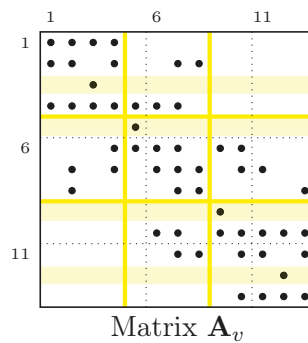


Abbildung 5.10.: Die Belegungsstrukturen der Matrix \mathbf{A}_v . Die Zeilen der betrachteten trockenen Knoten sind gelb unterlegt, während die gelben Linien die Stellen der Indexverschiebung markieren.

6. Gebietszerlegung

6.1. Partitionierung

Um die Tsunamisimulation zu beschleunigen, wird das räumliche Rechengebiet in p disjunkte Partitionen aufgeteilt. Die Anzahl der Partitionen entspricht dabei der Anzahl der verfügbaren Prozessoreinheiten (PE). Die Partitionierung wird im Folgenden mit Π_p bezeichnet.

6.1.1. Gebietszerlegung

Im hier verwendeten Ansatz werden die Partitionen über die Knotenmenge \mathcal{N} des Rechengitters definiert, siehe Abschnitt 3.1.1, so dass p disjunkte Knotenmengen

$$\mathcal{N}_k = \{n_{\pi_k(1)}, n_{\pi_k(2)}, \dots, n_{\pi_k(m_k)}\}, \quad k = 1, \dots, p \quad (6.1)$$

entstehen. Die Knotenmenge \mathcal{N}_k besteht aus m_k Gitterknoten $n_j^k = n_{\pi_k(j)}$, wobei der lokale Index $j = 1, \dots, m_k$ mit Hilfe der injektiven Indexfunktion π_k auf den globalen Index abgebildet wird. Für die Knotenmenge \mathcal{N} des gesamten Rechengebiets gilt

$$\mathcal{N} = \bigcup_{k=1}^p \mathcal{N}_k, \quad \mathcal{N}_i \cap \mathcal{N}_j = \emptyset \text{ für } i \neq j. \quad (6.2)$$

In Abbildung 6.1 ist im linken Bild eine solche Verteilung skizziert.

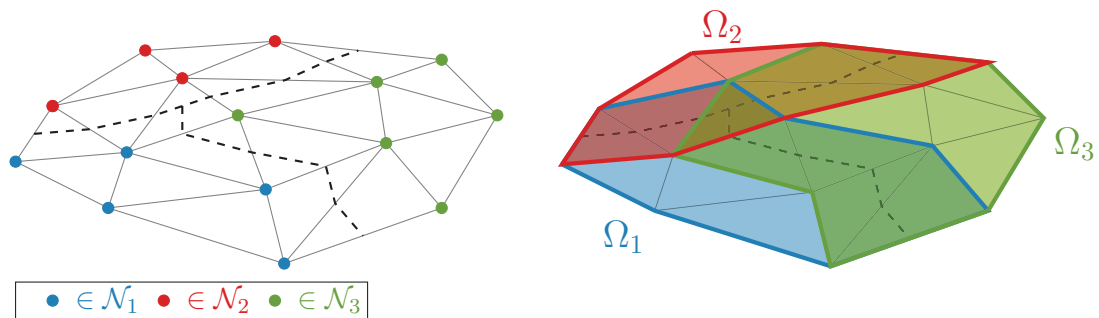


Abbildung 6.1.: Links: Zerlegung der Knotenmenge \mathcal{N} in disjunkte Partitionen \mathcal{N}_k .
Rechts: Zerlegung des Rechengebiets Ω in sich überlappende Teilgebiete Ω_k .

Da das Rechengitter nicht allein aus der Knotenmenge besteht, müssen auch die Kanten und Elemente der Triangulierung berücksichtigt werden. Diese werden jedoch

6. Gebietszerlegung

nicht explizit einer bestimmten Partition zugesprochen, da die zugehörigen Knoten in verschiedenen Partitionen liegen können. Elemente, die über Knoten verschiedener Partitionen definiert sind, werden im Folgenden Grenzelemente genannt. Die Kanten eines Grenzelements heißen Grenzkanten, die Knoten respektive Grenzknoten.

Jeder PE wird nun ein abgeschlossenes Teilgebiet $\Omega_k \subset \Omega$ zugeordnet, $k \in \{1, \dots, p\}$. Dieses Teilgebiet wird von der erweiterten Knotenmenge $\mathcal{N}_k^1 \supset \mathcal{N}_k$ aufgespannt. Die Erweiterung von \mathcal{N}_k erstreckt sich über die direkt angrenzenden Grenzknoten benachbarter Partitionen: Knoten die über mindestens einer Grenzkante mit \mathcal{N}_k verbunden sind. Die so definierten Teilgebiete Ω_k sind somit nicht disjunkt, sondern überlappen sich um die Breite eines Elements, wie im rechten Bild der Abbildung 6.1 graphisch dargestellt. Die Vereinigung der Teilgebiete überdeckt das gesamte Rechengitter und es gilt

$$\Omega = \bigcup_{k=1}^p \Omega_k, \quad \Omega_i \cap \Omega_j =: \Omega_{ij}. \quad (6.3)$$

Im Folgenden werden für jede PE die entsprechenden Knoten aus \mathcal{N}_k als lokale Knoten bezeichnet, während die Knoten aus $\mathcal{N}_k^1 \setminus \mathcal{N}_k$ externe Knoten genannt werden. Die übrigen Knoten aus $\mathcal{N} \setminus \mathcal{N}_k^1$ sind der einzelnen PE für gewöhnlich nicht bekannt.

Der Alternativansatz, statt der Knotenmenge das Rechengebiet disjunkt zu partitionieren, wird hier nicht verfolgt. Ursache dafür ist die Verteilung des linearen Gleichungssystems auf die verschiedenen PEs, welche aus der vorangehenden Gebietszerlegung resultiert. In Abschnitt 6.2 wird dieser Zusammenhang genauer beleuchtet. Matrix, Lösungsvektor und rechte Seite des Gleichungssystems sind über die Stützstellenmenge \mathcal{N} definiert, beziehen sich also auf die Gitterknoten. Durch die Partitionierung der Knotenmenge ist eine gleichmäßige Verteilung einfacher zu kontrollieren, da bei einer Zerlegung in disjunkte Teilgebiete Ω_k die Gitterknoten auf $\bar{\Omega}_i \cap \bar{\Omega}_k$ nicht automatisch eindeutig einer Partition zugeordnet werden.

6.1.2. Berechnung und Synchronisation

Sämtliche Werte, die auf einen Grenzknoten oder einen Grenzkantenmittelpunkt bezogen sind, werden entweder von verschiedenen PEs mehrfach berechnet oder synchronisiert. Dies hängt vom Träger der entsprechenden Ansatzfunktion ab. Befindet sich der Träger vollständig im Teilgebiet, so werden die Stützwerte berechnet. Andernfalls müssen die relevanten Werte von den zuständigen PEs der Nachbarpartitionen eingeholt werden. Der Datenaustausch wird dabei mit Hilfe von MPI-Kommunikationsroutinen realisiert.

Jeder auf \mathcal{N} gestützte Wert wird genau einmal berechnet und gegebenenfalls an die Nachbarpartition(en) versendet, falls es sich um einen Grenzknoten handelt. Auf Kantenmittelpunkten gestützte Werte werden nur dann synchronisiert, falls die Kante zwar zum Teilgebiet gehört, jedoch zwei externe Knoten verbindet. Verbindet die Kante jedoch Grenzknoten verschiedener Partitionen, so wird der entsprechende Wert von beiden PEs unabhängig bestimmt, wodurch die Synchronisation eingespart wird. In Abbildung 6.2 ist eine graphische Darstellung dieser Gegenüberstellung zu sehen.

Dabei wird die Anzahl der PEs, die einen Stützwert berechnen mit n_B und die Anzahl der Synchronisationen n_K bezeichnet. Es wird deutlich, dass bei Elementen die drei Teilgebieten angehören, n_K im Vergleich zu einfachen Grenzelementen erhöht wird, während n_B unverändert bleibt.

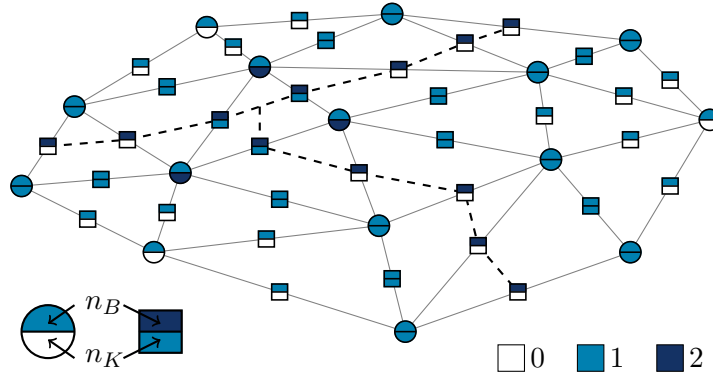


Abbildung 6.2.: Werte gestützt auf die Gitterknoten (Kreise) oder Kantenmittelpunkten (Quadrate) werden entsprechend der Lage innerhalb des Teilgebiets unterschiedlich oft berechnet oder synchronisiert.

6.1.3. Partitionierungssoftware

Es sind verschiedene kostenfreie Partitionierungsprogramme verfügbar, die trotz unterschiedlich strikter Lizenzierung alle die Nutzung für die hier aufgeführten Studien erlauben. Die Zerlegungen erfolgen nach Partitionierungsalgorithmen der Graphentheorie. Während METIS [34] mit einer Multilevel-Partitionierung den Graph $G_{\mathcal{T}}$ erst vergrößert, dann zerlegt und anschließend wieder entlang der berechneten Partitionen wieder verfeinert, verfolgt SCOTCH [48] mit dem Ansatz der rekursiven Bisektion das „Teile und Herrsche“-Prinzip. Die vorläufigen Partitionen einer groben Zerlegung werden weiter zerlegt und auf die so entstandenen Partitionen wird erneut der Zerlegungsalgorithmus angewandt. Dieses Verfahren wird solange wiederholt, bis die gewünschte Anzahl an Partitionen erreicht ist. Das Partitionierungsprogramm PaToH [13] vereint diese beiden Ansätze indem eine rekursive Bisektion auf dem größten Graph der Multilevel-Partitionierung angewendet wird. PaToH ist zudem das einzige dieser drei Programme, welches Hypergraphen zerlegt. Der Hypergraph ist ein spezieller Graph, dessen Verbindungen nicht ausschließlich zwei Knoten, sondern auch mehrere Knoten umfassen können.

Der Übergang von der Triangulierung \mathcal{T} zum Hypergraph $H_{\mathcal{T}}$ ist in Abbildung 6.3 skizziert. Die Verbindungen zwischen den Knoten stellen gerade die Tripel dar, welche die einzelnen Elemente Ω^e der Triangulierung aufspannen. Da die Berechnungen bei der FE Methode elementweise erfolgen, stellt der Hypergraph $H_{\mathcal{T}}$ die enge Beziehung der einzelnen Knoten zueinander besser dar als der Graph $G_{\mathcal{T}}$ aus Abbildung 5.1 es kann. Alle Partitionierungen der hier gerechneten Beispiele werden mit PaToH erzeugt.

6. Gebietszerlegung

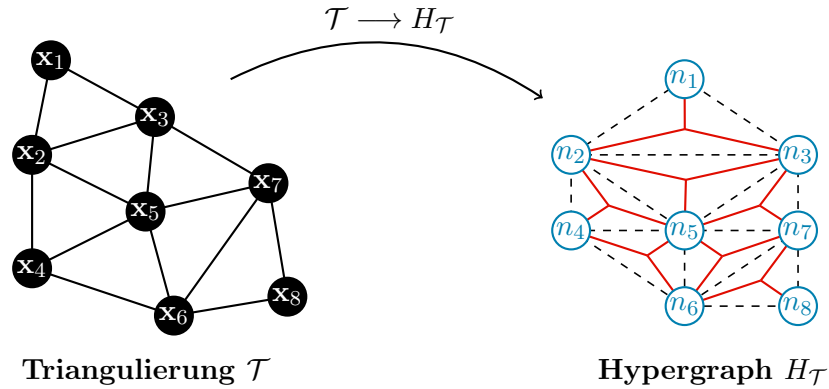


Abbildung 6.3.: Die N_e Verbindungen des Hypergraphs $H_{\mathcal{T}}$ beschreiben die Tripel an Gitterknoten, welche die einzelnen Elemente $\Omega^e \in \mathcal{T}$ aufspannen.

6.1.4. Gewichtung des Graphs

Das Ziel einer Graphpartitionierung ist, dass sich die Rechenlast der so entstehenden Partitionen im Gleichgewicht hält. Die Rechenlast besteht dabei aus Berechnung und Datenkommunikation. Dabei ist die Berechnung ein autonomer Prozess, während die Datenkommunikation Abhängigkeiten zwischen den PEs erzeugt, da diese gegebenenfalls aufeinander warten müssen. Die Graphpartitionierung versucht diese zwei Bestandteile über die einzelnen Partitionen hinweg auszubalancieren. Da vielleicht nicht an jedem Gitterknoten gleichviel gerechnet werden muss, oder hier und dort weniger Daten kommuniziert werden, besteht die Möglichkeit die Knoten und Verbindungen des Graphs zu gewichten. Die Gewichtung der Knoten beschreibt das Ausmaß an Berechnung, während die Gewichtung der Verbindungen das Ausmaß des Kommunikationsaufwands angibt. Da bei der \mathcal{P}^1 - \mathcal{P}_{NC}^1 Methode nicht alle Werte auf die Knoten gestützt sind, kann auch der Rechenaufwand an den Kanten in die Gewichtung der Verbindungen einfließen.

Bei einem ungewichteten Graph sind alle Knoten gleichwertig, so dass die Partitionen ungefähr die gleiche Anzahl an Knoten enthalten. Zudem ist jede Verbindung gleich teuer, so dass nicht berücksichtigt wird, welche Verbindungen zwischen benachbarter Partitionen geschnitten werden. Bei der Zerlegung des Hypergraphs $H_{\mathcal{T}}$ sind geschnittene Verbindungen gleichbedeutend mit den Grenzelementen des Rechengitters. In Abschnitt 9.4 wird der Einfluss verschiedener Partitionierungen untersucht. Dabei ist sehr entscheidend, ob in jedem Zeitschritt das Gesamtsystem oder das Teilsystem definiert über Ω_v gelöst wird.

6.2. Das verteilte Gleichungssystem

Bei Verwendung von Parallelrechnern mit verteiltem Speicher muss berücksichtigt werden, welche Daten pro PE lokal abgelegt sind, und respektive aus welchem Fremdspeicher darüberhinaus benötigte Daten angefordert werden müssen. Für das Lösen

eines Gleichungssystems $\mathbf{Ax} = \mathbf{b}$ bedeutet das, dass auch hier die Einträge von Matrix, Lösungsvektor und rechter Seite an verschiedenen Orten liegen. Dabei hängt die Verteilung dieser Werte sehr eng mit der gewählten Gebietszerlegung zusammen. Dieses wechselseitige Verhältnis wird hier nun näher erläutert.

Als Ausgangspunkt wird das Rechengitter Ω betrachtet. Nach der Durchführung einer Partitionierung für p PEs sind die disjunkten Mengen \mathcal{N}_k mit m_k Gitterknoten, $k = 1, \dots, p$ definiert. Zur Matrixassemblierung wird nun eine entsprechende globale Indizierung gewählt, so dass Knoten einer Partition aufeinanderfolgen. Erreicht wird diese Umsortierung durch die Permutation σ auf $\{1, \dots, n\}$, wobei

$$\sigma \circ \pi_k(i) = i + \sum_{j=1}^{k-1} m_j, \quad (6.4)$$

für alle $k = 1, \dots, p$ gilt. Abbildung 6.4 veranschaulicht eine solche Umsortierung.

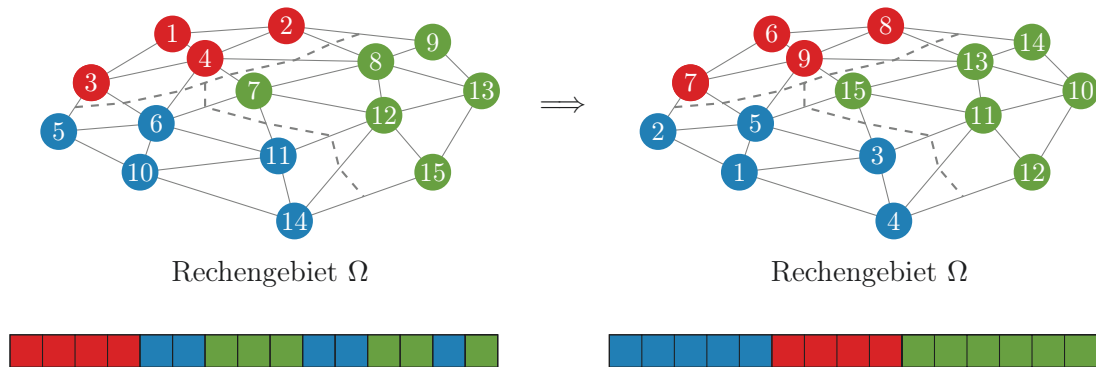


Abbildung 6.4.: Durch eine neue globale Indizierung enthält jede Matrix in der Nummerierung aufeinanderfolgende Knoten.

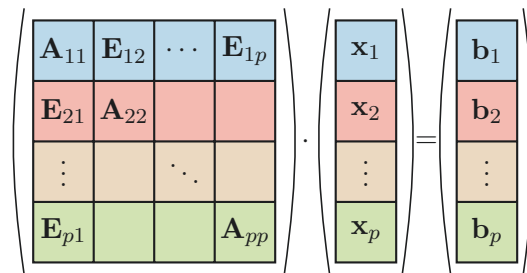


Abbildung 6.5.: Zeilenweise verteiltes Gleichungssystem. Die Partitionszugehörigkeit ist farblich markiert.

Nun werden alle Werte bezüglich einer Partition im Speicher der entsprechenden PE abgelegt. Dies gilt im Besonderen für die Matrixeinträge a_{ij} , die Werte x_i und b_i für $n_i \in \mathcal{N}_k$, $j = 1, \dots, n$. Die Matrixeinträge werden also zeilenweise verteilt, wie in Abbildung 6.5 dargestellt. Jede PE assembliert das entsprechende lokale Gleichungs-

6. Gebietszerlegung

system

$$\mathbf{A}_{kk}\mathbf{x}_k + \sum_{j \neq k} \mathbf{E}_{kj}\mathbf{x}_j = \mathbf{b}_k, \quad (6.5)$$

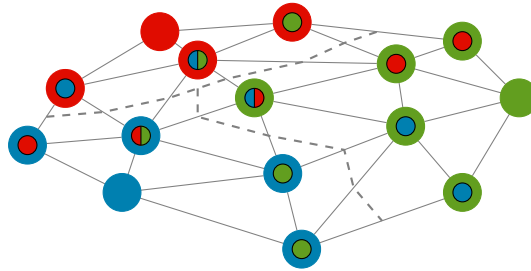
für $k \in \{1, \dots, p\}$, wobei die Blöcke \mathbf{E}_{kj} nur dann relevante Werte ungleich Null enthalten, falls $\mathcal{N}_k \cap \mathcal{N}_j^1 \neq \emptyset$. Die entsprechenden Knoten sind der PE als externe Knoten aus \mathcal{N}_k^1 bekannt.

6.2.1. Lokale Umsortierung

Die lokalen Knotenengen \mathcal{N}_k , $k = 1, \dots, p$ können nun intern klassifiziert werden, indem zwischen Grenzknoten und inneren Knoten unterschieden wird. Die Knotenmengen

$$\mathcal{Q}_k := \bigcup_{j \neq k} \mathcal{Q}_{kj} \subset \mathcal{N}_k \quad \text{mit} \quad \mathcal{Q}_{kj} := \mathcal{N}_k \cap \mathcal{N}_j^1 \quad (6.6)$$

enthalten demnach gerade die m_k^g lokalen Grenzknoten, während $\mathcal{N}_k \setminus \mathcal{Q}_k$ die m_k^i lokalen inneren Knoten umfassen. Jeder lokale Knoten entspricht genau einer dieser Teilmengen, so dass die m_k^i inneren und m_k^g Grenzknoten gemeinsam die $m_k = m_k^i + m_k^g$ Knoten von \mathcal{N}_k bilden. In Abbildung 6.6 ist eine solche Klassifizierung skizziert.



Rechengebiet Ω

Abbildung 6.6.: Klassifizierung der lokalen Knoten in innere Knoten (einfarbig) und Grenzknoten (mehrfarbig). Farbgebung: außen - Partitionszugehörigkeit, innen - Nachbarpartition(en)

Die lokale Indizierung von \mathcal{N}_k^1 wird nun so gewählt, dass die Indizes 1 bis m_k^i die inneren Knoten beschreiben, $m_k^i + 1$ bis m_k die lokalen Grenzknoten und $m_k + 1$ bis m_k^e die Grenzknoten benachbarter Partitionen. Diese Anordnung ist grundlegend im Zusammenhang mit Vorkonditionierern auf der Basis des Schurkomplements, wie in Abschnitt 8.3.1 näher erläutert wird. Unter Berücksichtigung der Nachbarschaftsbeziehungen der Grenzknoten, können die lokalen und externen Grenzknoten weiter gruppiert und umsortiert werden, wie in Abbildung 6.7 symbolisiert. Da gerade die auf Grenzknoten bezogenen Daten zwischen den PEs ausgetauscht werden, erleichtert diese blockartige Zusammensetzung die Datensynchronisation. Eine analoge Umsortierung wird für die Grenzkanten und -elemente durchgeführt.

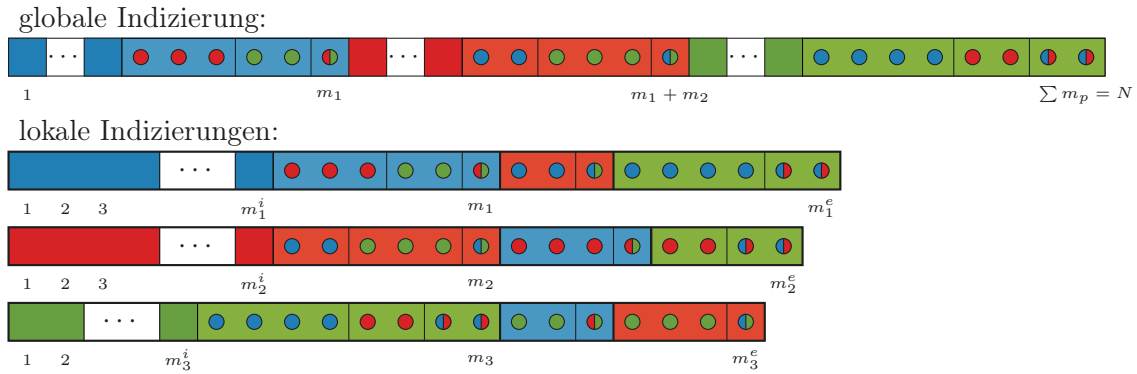


Abbildung 6.7.: Die Zusammensetzung der lokalen Knotennummerierung erfolgt blockweise.

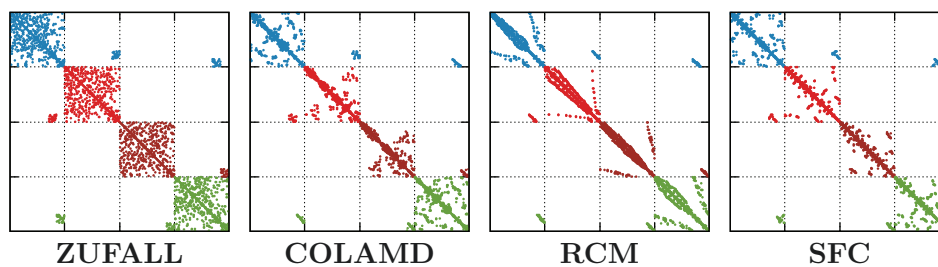


Abbildung 6.8.: Das Füllmuster der verteilten Matrix weist charakteristische Strukturen der globalen Sortierung auf.

Da die Knoten entlang der globalen Nummerierung des nichtverteilten Systems wie in Abbildung 6.4 in die entsprechenden Klassen einsortiert werden, bleibt gerade in dem relativ großen Block, der sich auf die Interaktionen der inneren Knoten bezieht, die globale Sortierung weitgehend erhalten. Die Füllstrukturen der verteilten Matrizen zeigen weiterhin charakteristische Muster der Knotennummerierung auf, wie in Abbildung 6.8 zu sehen ist. Die konzentrierte Füllstruktur der Blöcke außerhalb der Diagonalen wird durch die Gruppierungen von inneren und Grenzknoten herbeigeführt.

6.2.2. Gesamt- oder reduziertes System

Die Möglichkeiten in jedem Zeitschritt das Gesamtsystem zu lösen, oder sich das auf Ω_v reduzierte System zu beschränken, wie in Abschnitt 5.3 erläutert, bringen im Fall eines verteilten Systems neue Aspekte mit ein. Die Berechnung des Gesamtsystems hat den Vorteil, dass durch eine geschickte Gebietszerlegung eine gleichmäßige Verteilung der Matrix auf die verschiedenen PEs erreicht werden kann. Da jedoch die Lösung an den trockenen Knoten von vornherein feststeht, ist dieses Gleichgewicht nicht automatisch durch eine gleichmäßige Knotenanzahl pro PE erreicht. Besitzt eine Partition nur trockene Knoten, so ist die Konvergenzrate in der zugehörigen Teilmatrix

7. Krylov-Unterraumverfahren

Für die Berechnung des nichthydrostatischen Bodendrucks in TsunAWI-NH muss ein Gleichungssystem gelöst werden, dessen reguläre Matrix \mathbf{A} groß, dünnbesetzt und unsymmetrisch ist. Direkte Lösungsverfahren wie etwa die Gauß-Elimination sind in diesem Fall eher ungeeignet, da die dünnbesetzte Struktur nicht optimal ausgenutzt werden kann und sich der Speicherbedarf vervielfacht. Deshalb wird die Aufmerksamkeit der iterativen Lösungsmethoden gewidmet, genauer gesagt, der Klasse der Krylov-Unterraumverfahren. Einen Überblick über die Entwicklung der iterativen Lösungsverfahren ist in [57] beschrieben.

7.1. Iteratives Lösungsverfahren

Bei einem iterativen Lösungsansatz wird ein Gleichungssystem

$$\mathbf{Ax} = \mathbf{b} \quad (7.1)$$

mit $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ gelöst, indem eine Vektorenfolge $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m)$ mit $\mathbf{x}_i \in \mathbb{R}^n$, konstruiert wird, die gegen den Lösungsvektor \mathbf{x} strebt. Da die Differenz $\mathbf{d}_i = \mathbf{x} - \mathbf{x}_i$ zwischen der tatsächlichen Lösung \mathbf{x} und dem approximierten Lösungsvektor \mathbf{x}_i unbekannt ist, wird als vergleichbare Größe das Residuum

$$\mathbf{r}_i := \mathbf{b} - \mathbf{Ax}_i \quad (7.2)$$

in die Konstruktion der Vektorenfolge mit einbezogen. Das Residuum beschreibt die Abweichung $\mathbf{A}(\mathbf{x} - \mathbf{x}_i) = \mathbf{Ad}_i$. Beginnend mit einem beliebigen Startvektor $\mathbf{x}_0 \in \mathbb{R}^n$ wird mit Algorithmus 7.1 basierend auf der Richardson-Iteration die Vektorfolge generiert.

Algorithmus 7.1: Iteratives Lösungsverfahren basierend auf der Richardson-Iteration.

```
 $\mathbf{r}_0 = \mathbf{b} - \mathbf{Ax}_0$ 
for ( $i = 0, \|\mathbf{r}_i\|_2 > \varepsilon_{\text{tol}} \ \& \ i \leq i_{\text{max}}, i++$ )
|    $\mathbf{x}_{i+1} := \mathbf{x}_i + \mathbf{r}_i$ 
|    $\mathbf{r}_{i+1} := \mathbf{b} - \mathbf{Ax}_{i+1}$ 
```

Statt der einmaligen Matrix-Vektor-Multiplikation mit der Inversen \mathbf{A}^{-1} , deren Berechnung meist sehr rechenintensiv ausfällt, wird pro Iteration eine Matrix-Vektor-Multiplikation mit der bekannten, dünnbesetzten Matrix \mathbf{A} getätigt. Der Algorithmus

7. Krylov-Unterraumverfahren

stoppt, wenn die Residuumsnorm eine vorher gesetzte Grenze ε_{tol} unterschreitet. Da die Vektorenfolge nicht zwingend konvergiert, wird zur Sicherheit eine maximale Anzahl von Iterationsschritten i_{max} festgelegt. Ist nach i_{max} Iterationsschritten die Norm des Residuums größer als die gewünschte Toleranz, so wird das Verfahren (erfolglos) abgebrochen.

Ob und wie schnell die konstruierte Vektorenfolge gegen den Lösungsvektor \mathbf{x} konvergiert, hängt stark von den Eigenschaften der Matrix \mathbf{A} ab. Aus der Vorgehensweise nach Algorithmus 7.1 folgt die iterative Beschreibung

$$\mathbf{r}_{i+1} = \mathbf{b} - \mathbf{A}\mathbf{x}_{i+1} = (\mathbf{I} - \mathbf{A})\mathbf{r}_i \quad (7.3)$$

des Residuums. Mit der daraus resultierenden Abschätzung

$$\|\mathbf{r}_{i+1}\| \leq \|\mathbf{I} - \mathbf{A}\| \|\mathbf{r}_i\|, \quad (7.4)$$

strebt die Vektorenfolge $\mathbf{x}_i \in \mathbb{R}^n$, $i = 1, 2, \dots$ gegen \mathbf{x} , falls $\|\mathbf{I} - \mathbf{A}\| < 1$. Erfüllt \mathbf{A} diese Voraussetzung nicht, so muss das Gleichungssystem vorkonditioniert werden.

7.2. Lösungsraum

Ein Vektor \mathbf{x}_i der konstruierten Vektorenfolge kann explizit in der Form

$$\mathbf{x}_i = \mathbf{x}_0 + \sum_{j=0}^{i-1} \mathbf{r}_j, \quad \text{mit} \quad \mathbf{r}_j = (\mathbf{I} - \mathbf{A})^j \mathbf{r}_0 \quad (7.5)$$

dargestellt werden. Der Vektor \mathbf{x}_i liegt demnach im Raum $\mathbf{x}_0 + \mathcal{K}^i(\mathbf{A}; \mathbf{r}_0)$, wobei

$$\mathcal{K}^i(\mathbf{A}; \mathbf{r}_0) = \text{span}\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \mathbf{A}^2\mathbf{r}_0, \dots, \mathbf{A}^{i-1}\mathbf{r}_0\}. \quad (7.6)$$

als Krylov-Unterraum bezeichnet wird. Falls nicht anders gekennzeichnet wird im Folgenden mit \mathcal{K}^i stets $\mathcal{K}^i(\mathbf{A}; \mathbf{r}_0)$ beschrieben.

Als Krylov-Unterraumverfahren werden iterative Methoden bezeichnet, in denen der Lösungsvektor \mathbf{x} durch eine Vektorenfolge ($\mathbf{x}_i \in \mathbf{x}_0 + \mathcal{K}^i$) angenähert wird. Hierbei gibt es mehrere Ansätze für die Konstruktion der Basis des Krylov-Unterraums sowie der Vektorenfolge \mathbf{x}_i . Da für die Berechnung des dynamischen Bodendrucks die Matrix des Gleichungssystem (3.22) unsymmetrisch ist, ist die Wahl des Verfahrens eingeschränkt. Die hier verwendete Methode wurde von Saad und Schultz in [55] unter dem Namen Generalized Minimum Residual Algorithmus (GMRES) für allgemeine lineare Gleichungssysteme vorgestellt. Dabei wird jener Vektor aus $\mathbf{x}_i \in \mathbf{x}_0 + \mathcal{K}^i$ in die Vektorenfolge aufgenommen, dessen Residuum $\mathbf{r}_i \in \mathcal{K}^{i+1}$ bezüglich der euklidischen Standardnorm $\|\cdot\|_2$ minimal ist:

$$\min_{\mathbf{x}_i \in \mathbf{x}_0 + \mathcal{K}^i} \|\mathbf{b} - \mathbf{A}\mathbf{x}_i\|_2. \quad (7.7)$$

Dieses Verfahren hat den Vorteil, dass der Algorithmus genau dann abbricht, wenn \mathbf{x}_m das exakte Ergebnis beschreibt [55]. Das häufig verwendete Krylov-Unterraumverfahren BiCGStab [70] für unsymmetrische lineare Gleichungssysteme,

das Ansätze des bikonjugierten Gradientenverfahren BiCG [20] und des GMRES-Algorithmus vereint, hat sich in Voruntersuchungen für die nichthydrostatische Tsunamisimulation mit Überflutung aufgrund von langsamer Konvergenz als ungeeignet erwiesen. Da beim vorkonditionierten BiCGStab-Algorithmus zwei Prädiktionierungsoperationen pro Iterationsschritt stattfinden, sind Prädiktionierungsmethoden mit Datenaustausch sehr teuer in der Anwendung. Somit wird die Rechenzeit entweder durch viele Iterationsschritte bei schlechter Konvergenzrate oder durch zeitintensive Vorkonditionierung bestimmt. Die Untersuchungen in dieser Arbeit beschränken sich deshalb auf den GMRES-Ansatz als Lösungsverfahren.

7.3. Basisvektoren

Die intuitive Basis $\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \dots, \mathbf{A}^{m-1}\mathbf{r}_0\}$ des Krylov-Unterraums \mathcal{K}^m eignet sich nicht zum Aufbau einer Vektorenfolge: Starke Eigenvektoren von \mathbf{A} agieren richtungsdominant auf $\mathbf{A}^k\mathbf{r}_0$, $1 \leq k \leq m-1$. Im endlichen Zahlenraum führt das zum Verlust der linearen Unabhängigkeit [19]. Stattdessen wird ausgehend von $\mathbf{v}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2$ eine orthonormale Basis $\mathcal{V}_m = \{\mathbf{v}_1, \dots, \mathbf{v}_m\}$ aufgebaut. In [55] wird die Basis der Krylov-Unterräume mittels Arnoldi-Methode und modifiziertem Gram-Schmidt-Verfahren (mGS) konstruiert. Die Verwendung von Householder-Transformationen (HT) für die Konstruktion der Basis im GMRES-Algorithmus zeigte [71]. Dieses Vorgehen erfordert mehr Arithmetik als mGS, liefert jedoch bessere Ergebnisse in orthogonalisierungskritischen Fällen [25]. Auch können kollektive Kommunikationsaufrufe in der parallelen Berechnung umgangen werden, indem globale Householder-Operatoren durch Kombinationen von lokalen Householder-Transformationen und globalen Givens-Rotationen ersetzt werden [61]. Diese alternative Vorgehensweise der HT-generierten Krylovbasis wurde nach Algorithmen aus [25] und [61] implementiert. Doch zeigten Voruntersuchungen, dass bei den hier untersuchten Rechenbeispielen der erhöhte Rechenaufwand überwiegt, so dass auch bei der parallelen Berechnung kein Vorteil in Bezug auf Effizienz im Vergleich zu mGS zu verzeichnen ist. In Kapitel 9 wird deshalb ausschließlich die mGS-erzeugte Basis aus Algorithmus 7.2 verwendet.

Algorithmus 7.2: Der modifizierte Gram-Schmidt-Algorithmus im Pseudo-Code.

```

for ( $j = 1, j \leq m, j++$ )
     $\mathbf{w} := \mathbf{A}\mathbf{v}_j$ 
    for ( $i = 1, i \leq j, i++$ )
         $h_{ij} := \langle \mathbf{w}, \mathbf{v}_i \rangle$ 
         $\mathbf{w} := \mathbf{w} - h_{ij}\mathbf{v}_i$ 
     $h_{j+1,j} := \|\mathbf{w}\|_2$ 
    if ( $h_{j+1,j} == 0$ ) stop
     $\mathbf{v}_{j+1} := \mathbf{w}/h_{j+1,j}$ 

```

7. Krylov-Unterraumverfahren

In Matrixschreibweise kann die Beziehung der Basisvektoren aus \mathcal{V}_m und \mathcal{V}_{m+1} der Krylov-Unterräume \mathcal{K}^m und \mathcal{K}^{m+1} mit

$$\mathbf{A}\mathbf{V}_m = \mathbf{V}_{m+1}\mathbf{H}_m, \quad (7.8)$$

zusammengefasst werden, wobei $\mathbf{H}_m \in \mathbb{R}^{m+1 \times m}$ die obere Hessenbergmatrix mit den Einträgen h_{ij} aus Algorithmus 7.2 beschreibt und \mathbf{V}_m die Basisvektoren aus \mathcal{V}_m als Spaltenvektoren besitzt.

7.4. Verfahrensprinzip

Beschreibt $\mathcal{V}_m = \{\mathbf{v}_1, \dots, \mathbf{v}_m\}$ die Orthonormalbasis des Krylov-Unterraums \mathcal{K}^m , so kann jeder Vektor $\mathbf{x}_m \in \mathbf{x}_0 + \mathcal{K}^m$ mittels

$$\mathbf{x}_m = \mathbf{x}_0 + \mathbf{V}_m\mathbf{y}, \quad (7.9)$$

und das zugehörige Residuum \mathbf{r}_m mit

$$\mathbf{r}_m = \mathbf{b} - \mathbf{A}\mathbf{x}_m = \mathbf{b} - \mathbf{A}\mathbf{x}_0 - \mathbf{A}\mathbf{V}_m\mathbf{y} = \mathbf{r}_0 - \mathbf{A}\mathbf{V}_m\mathbf{y} \quad (7.10)$$

als Linearkombinationen dargestellt werden. Mit $\mathbf{v}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2$ und der Beziehung (7.8) kann das Residuum auch mit

$$\mathbf{r}_m = \|\mathbf{r}_0\|_2 \mathbf{v}_1 - \mathbf{V}_{m+1}\mathbf{H}_m\mathbf{y} = \mathbf{V}_{m+1}(\|\mathbf{r}_0\|_2 \mathbf{e}_1 - \mathbf{H}_m\mathbf{y}), \quad (7.11)$$

beschrieben werden.

Nun wird mittels einer orthonormalen Matrix $\mathbf{Q}_m \in \mathbb{R}^{m+1 \times m+1}$ die obere Hessenbergmatrix \mathbf{H}_m in eine obere Dreiecksmatrix $\mathbf{R}_m \in \mathbb{R}^{m+1 \times m}$ übergeführt:

$$\mathbf{Q}_m\mathbf{H}_m = \mathbf{R}_m. \quad (7.12)$$

Die letzte Zeile der Dreiecksmatrix $\mathbf{R}_m \in \mathbb{R}^{m+1 \times m}$ entspricht dabei einer Nullzeile. Da nur eine geringe Anzahl an Einträgen eliminiert werden muss, eignet sich hierfür das Givens-Verfahren. Dabei werden durch eine Hintereinanderausführung von Drehmatrizen $\mathbf{G}_{i+1,i}$ nacheinander die Einträge $h_{i+1,i}$, $1 \leq i \leq m$ auf der unteren Nebendiagonalen von \mathbf{H}_m ausgelöscht. Durch die Orthonormalität der Matrix \mathbf{Q}_m gilt $\mathbf{Q}_m^T = \mathbf{Q}_m^{-1}$ und das Residuum aus (7.11) erfüllt

$$\mathbf{r}_m = \mathbf{V}_{m+1}\mathbf{Q}_m^T(\underbrace{\|\mathbf{r}_0\|_2 \mathbf{Q}_m \mathbf{e}_1}_{=: \mathbf{g}_m \in \mathbb{R}^{m+1}} - \mathbf{R}_m\mathbf{y}). \quad (7.13)$$

Die Minimierungsaufgabe (7.7), welche die Konstruktion der Vektorenfolge mit sich bringt, kann mit (7.13) zu

$$\min_{\mathbf{x}_m \in \mathbf{x}_0 + \mathcal{K}^m} \|\mathbf{b} - \mathbf{A}\mathbf{x}_m\|_2 = \min_{\mathbf{y} \in \mathbb{R}^m} \|\mathbf{g}_m - \mathbf{R}_m\mathbf{y}\|_2 \quad (7.14)$$

umformuliert werden, da die Orthonormalmatrizen \mathbf{V}_{m+1} und \mathbf{Q}_m^T aus (7.13) keinen Einfluss auf die Norm ausüben. Die Nullzeile in \mathbf{R}_m verhindert jeglichen Einfluss von \mathbf{y} auf g_{m+1} , den $(m+1)$ -ten Eintrag des Vektors \mathbf{g}_m , siehe Abbildung 7.1. Deshalb kann die Minimierungsaufgabe auf die ersten m Zeilen reduziert werden und das Minimum wird mit dem Vektor \mathbf{y} erreicht, für den

$$\mathbf{R}'_m \mathbf{y} = \mathbf{g}'_m, \quad (7.15)$$

gilt. Hierbei beschreiben $\mathbf{g}'_m \in \mathbb{R}^m$ und $\mathbf{R}'_m \in \mathbb{R}^{m \times m}$ die ersten m Zeilen von \mathbf{g}_m beziehungsweise \mathbf{R}_m . Die Gleichung (7.15) kann durch Rücksubstitution nach \mathbf{y} aufgelöst werden.

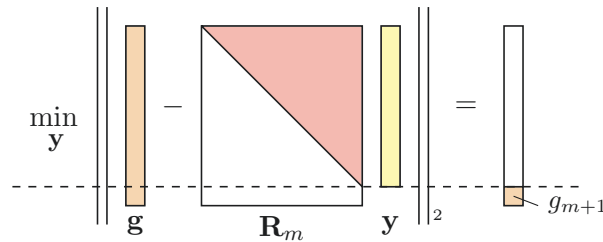


Abbildung 7.1.: Schematische Darstellung der Minimierungsaufgabe von GMRES. Nulleinträge sind nicht eingefärbt.

Tatsächlich werden die Vektoren \mathbf{y} und \mathbf{x}_m in den einzelnen Iterationsschritten nicht berechnet, da sie nicht für den nächsten Schritt benötigt werden. Dies spart viele unnötigen Rechenoperationen ein. Nach Bestimmung der Givensrotation $\mathbf{G}_{m+1,m}$ beschränkt sich die Berechnung auf den Vektor

$$\mathbf{g}_m := \|\mathbf{r}_0\|_2 \mathbf{Q}_m \mathbf{e}_1 = \mathbf{G}_{m+1,m} \begin{pmatrix} \mathbf{g}^{m-1} \\ 0 \end{pmatrix}. \quad (7.16)$$

Auch ohne das zugehörige Residuum $\mathbf{r}_m = \mathbf{V}_{m+1} \mathbf{Q}_m^T (0, \dots, 0, g_{m+1})^T$ explizit zu berechnen, beträgt dessen Norm

$$\|\mathbf{r}_m\|_2 = |g_{m+1}|. \quad (7.17)$$

Sofern das Verfahren konvergiert, wird im praktischen Fall das Iterationsverfahren abgebrochen, wenn g_{m+1} betragsmäßig die gesetzte Toleranzgrenze ε_{tol} unterschreitet und die Genauigkeit des Approximationsvektors \mathbf{x}_m genügt. Erst dann werden mit (7.15) die Koeffizienten für die Linearkombination in (7.9) bestimmt und die Lösung \mathbf{x}_m berechnet. Für ein Gleichungssystem mit $\mathbf{A} \in \mathbb{R}^{n \times n}$, und $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ wird mit dem GMRES-Verfahren spätestens nach n Iterationen die Lösung \mathbf{x} bestimmt, da die Krylov-Unterräume Teilräume von \mathbb{R}^n sind und somit nicht mehr als n Dimensionen haben können [55].

7.5. GMRES mit Vorkonditionierung

Um die Konvergenzrate zu beschleunigen beziehungsweise die Konvergenz zu ermöglichen, wird das Verfahren vorkonditioniert. Es wird ein Gleichungssystem gelöst, das dem

7. Krylov-Unterraumverfahren

eigentlichen System (7.1) ähnelt, jedoch besser konditioniert und einfacher zu lösen ist. Auf diesem Wege kann die Lösung von (7.1) schneller ermittelt werden. An dieser Stelle wird die Kombination des GMRES-Verfahrens mit den allgemeinen Ansätzen der links- und rechtsseitigen Vorkonditionierung kurz vorgestellt.

Im Falle einer linksseitigen Vorkonditionierung wird statt der (7.1) das Gleichungssystem

$$\mathbf{K}^{-1}\mathbf{A}\mathbf{x} = \mathbf{K}^{-1}\mathbf{b} \quad (7.18)$$

analog mit dem GMRES-Verfahren gelöst, wobei \mathbf{K}^{-1} den Prädiktionierungsoperator beschreibt. Dabei wird das vorkonditionierte Residuum

$$\hat{\mathbf{r}}_i = \mathbf{K}^{-1}(\mathbf{b} - \mathbf{A}\mathbf{x}_i) \quad (7.19)$$

mit $\hat{\mathbf{r}}_i \in \mathcal{K}^{i+1}(\mathbf{K}^{-1}\mathbf{A}, \hat{\mathbf{r}}_0)$ im Krylov-Unterraum $\mathbf{x}_0 + \mathcal{K}^i(\mathbf{K}^{-1}\mathbf{A}, \hat{\mathbf{r}}_0)$ minimiert. Hierbei muss beachtet werden, dass möglicherweise das vorkonditionierte Residuum $\hat{\mathbf{r}}_i$ schneller konvergiert als das \mathbf{r}_i bezogen auf (7.1). In diesem Fall bricht die Methode zu früh ab und die Annäherung der Vektorenfolge (\mathbf{x}_i) an die Lösung \mathbf{x} ist unvollständig.

Algorithmus 7.3: Der GMRES-Algorithmus mit links- und rechtsseitiger Vorkonditionierung.

linksseitige Vorkonditionierung	rechtsseitige Vorkonditionierung
<pre> 1 $\mathbf{r}_0 := \mathbf{K}^{-1}(\mathbf{b} - \mathbf{A}\mathbf{x}_0); \quad \rho := \ \mathbf{r}_0\ _2$ 2 $\mathbf{v}_1 := \mathbf{r}_0/\rho; \quad \mathbf{g} := \rho \mathbf{e}_1$ 3 for ($i = 1; \rho > \varepsilon_{tol}; i++$) 4 $\mathbf{z} := \mathbf{A}\mathbf{v}_i$ 5 $\mathbf{w} := \mathbf{K}^{-1}\mathbf{z}$ 6 $[\mathbf{v}_{i+1}, \mathbf{R}_i, \mathbf{g}] := \text{krylov}(\mathbf{w}, \mathbf{R}_{i-1}, \mathbf{g})$ 7 $\rho := g_{i+1}$ 8 $\mathbf{y}_i := \mathbf{R}_i'^{-1}\mathbf{g}'$ 9 $\mathbf{x}_i := \mathbf{x}_0 + \mathbf{V}_i\mathbf{y}_i$ </pre>	<pre> $\mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0; \quad \rho := \ \mathbf{r}_0\ _2$ $\mathbf{v}_1 := \mathbf{r}_0/\rho; \quad \mathbf{g} := \rho \mathbf{e}_1$ for ($i = 1; \rho > \varepsilon_{tol}; i++$) $\mathbf{z} := \mathbf{K}^{-1}\mathbf{v}_i$ $\mathbf{w} := \mathbf{A}\mathbf{z}$ $[\mathbf{v}_{i+1}, \mathbf{R}_i, \mathbf{g}] := \text{krylov}(\mathbf{w}, \mathbf{R}_{i-1}, \mathbf{g})$ $\rho := g_{i+1}$ $\mathbf{y}_i := \mathbf{R}_i'^{-1}\mathbf{g}'$ $\mathbf{x}_i := \mathbf{x}_0 + \mathbf{K}^{-1}\mathbf{V}_i\mathbf{y}_i$ </pre>

Bei der rechtsseitiger Vorkonditionierung wird iterativ das Gleichungssystem

$$\mathbf{A}\mathbf{K}^{-1}\hat{\mathbf{x}} = \mathbf{b}, \quad \text{für } \mathbf{x} = \mathbf{K}^{-1}\hat{\mathbf{x}} \quad (7.20)$$

gelöst. Die im GMRES-Verfahren generierten Basisvektoren aus \mathcal{V}_i spannen den entsprechenden Krylov-Unterraum $\mathcal{K}^i(\mathbf{A}\mathbf{K}^{-1}, \mathbf{r}_0)$ auf. Für das Residuum des vorkonditionierten Systems kann mit Gleichung (7.22) die Abhängigkeit

$$\mathbf{r}_i = \mathbf{b} - \mathbf{A}\mathbf{K}^{-1}\hat{\mathbf{x}}_i = \mathbf{b} - \mathbf{A}\mathbf{x}_i, \quad (7.21)$$

aufgezeigt werden. Die Residuen der Gleichungen (7.1) und (7.20) stimmen überein, so dass auch bei rechtsseitiger Vorkonditionierung die Minimierungsaufgabe (7.14)

zur Konstruktion der Vektorenfolge herangezogen wird [54]. Bei Iterationsende nach m Schritten kann der Lösungsvektor \mathbf{x} mit

$$\mathbf{x} = \mathbf{x}_0 + \mathbf{K}^{-1}\mathbf{V}_m\mathbf{y} \quad (7.22)$$

bestimmt werden. Die vorkonditionierten GMRES-Implementierungen sind im Pseudo-Code in Algorithmus 7.3 dargestellt. Die Unterschiede von links- und rechtsseitiger Vorkonditionierung sind dabei grün hervorgehoben. Die Anzahl der Rechenoperationen ist bei gleicher Anzahl von Iterationsschritten identisch, ebenso wie die Speicheranforderungen.

Bei der Vorkonditionierung werden häufig auch unvollständige Lösungsmethoden verwendet, welche die Lösung approximieren statt zu berechnen. Darauf wird in Kapitel 8 genauer eingegangen. Wird eine iterative Methode verwendet, so existiert kein konstanter Präkonditionierungsoperator \mathbf{K}^{-1} . Ausgehend von GMRES mit rechtsseitiger Vorkonditionierung wurde in [53] unter dem Namen Flexible GMRES (FGMRES) eine Variante präsentiert, die den Gebrauch von variablen Vorkonditionierungsoperatoren erlaubt. Beim FGMRES-Ansatz kann statt eines konstanten Operators \mathbf{K}^{-1} auch eine Folge von Operatoren (\mathbf{K}_i^{-1}) verwendet werden. Neben den Basisvektoren \mathbf{v}_i werden die vorkonditionierten Vektoren

$$\mathbf{z}_i = \mathbf{K}_i^{-1}\mathbf{v}_i, \quad 1 \leq i \leq m \quad (7.23)$$

gespeichert, vergleiche Zeile 4 in Algorithmus 7.3 für das rechtsseitig vorkonditionierte GMRES-Verfahren. In Gleichung (7.14) wird die minimale Residuumsnorm nicht im Raum $\mathbf{x}_0 + \mathcal{K}^m(\mathbf{A}\mathbf{K}^{-1}, \mathbf{r}_0)$ sondern in $\mathbf{x}_m \in \mathbf{x}_0 + \text{span}(\mathcal{Z}_m)$ gesucht, wobei \mathcal{Z}_m die Vektoren $\mathbf{z}_i \in \mathcal{Z}_m$ enthält. Entsprechend erfüllen die Basisvektoren statt (7.8) die Gleichung

$$\mathbf{A}\mathbf{Z}_m = \mathbf{V}_{m+1}\mathbf{H}_m, \quad (7.24)$$

und das Residuum

$$\mathbf{r}_m = \mathbf{b} - \mathbf{A}\mathbf{z}_m = \mathbf{r}_0 - \mathbf{A}\mathbf{Z}_m\mathbf{y} = \mathbf{r}_0 - \mathbf{V}_{m+1}\mathbf{H}_m\mathbf{y} \quad (7.25)$$

kann weiterhin in Gleichung (7.11) überführt werden [53]. Endet der Algorithmus nach m Iterationsschritten wird der Lösungsvektor \mathbf{x} dann mit

$$\mathbf{x} = \mathbf{x}_0 + \mathbf{Z}_m\mathbf{y}, \quad (7.26)$$

berechnet. Dabei besteht die Matrix \mathbf{Z}_m aus den Spaltenvektoren $\mathbf{z}_i \in \mathcal{Z}_m$. Bricht das FGMRES-Verfahren nach m Iterationsschritten ab, so beschreibt \mathbf{x}_m genau dann die exakte Lösung \mathbf{x} , falls \mathbf{H}'_m regulär ist [53].

Da im GMRES-Verfahren sowohl die Matrizen \mathbf{V}_m und \mathbf{H}_m , bei FGMRES zusätzlich noch die Matrix \mathbf{Z}_m gespeichert werden müssen, kann dies bei höherer Iterationsanzahl zu sehr hohen Speicheranforderungen führen. Um den Speicherbedarf zu senken, werden die Gleichungen (7.15) und (7.9) nach $i_m < i_{\max}$ Iterationsschritten gelöst, und der Algorithmus mit $\mathbf{x}_0 = \mathbf{x}_{i_m}$ neugestartet, wie in [55] unter dem Namen GMRES(m) beschrieben. Somit kann der benötigte Speicher kontrolliert werden, auch wenn dies erhöhten Rechenaufwand bedeutet und das Verfahren möglicherweise langsamer oder gar nicht konvergiert, da die Krylov-Unterräume nur einen Teil des \mathbb{R}^n abdecken.

8. Präkonditionierungsmodelle

Das Krylov-Unterraumverfahren aus Kapitel 7 muss vorkonditioniert werden, falls die Matrix \mathbf{A} die notwendigen spektralen Anforderungen nicht erfüllt. Mit einer geeigneten Präkonditionierungsmethode kann zudem die Konvergenzrate beschleunigt werden. Statt des Gleichungssystems (7.1) wird dann das System (7.18) beziehungsweise (7.20) gelöst. Bei der Wahl der Vorkonditionierungsmethode wird vorausgesetzt, dass der Präkonditionierungsoperator \mathbf{K}^{-1} einfach zu berechnen ist und $\mathbf{K}^{-1} \approx \mathbf{A}^{-1}$ gilt. Bei Gleichheit gilt $\mathbf{AK}^{-1} = \mathbf{K}^{-1}\mathbf{A} = \mathbf{I}$, wobei $\mathbf{I} \in \mathbb{R}^{n \times n}$ die (hervorragend konditionierte) Einheitsmatrix darstellt. Die Motivation der Präkonditionierung ist also ohne viel Aufwand ein Gleichungssystem zu schaffen, das ähnlich gute Konvergenzeigenschaften wie die Einheitsmatrix besitzt, wodurch das eigentliche Gleichungssystem leicht gelöst werden kann. Es gibt viele Methoden, bei denen \mathbf{K}^{-1} nicht explizit aufgestellt wird. Stattdessen wird die Lösung eines Gleichungssystems der Form

$$\mathbf{Ax} = \mathbf{b} \quad (8.1)$$

für $\mathbf{A} \in \mathbb{R}^{n \times n}$ und $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ angenähert, da aus $\mathbf{x} = \mathbf{K}^{-1}\mathbf{b} \approx \mathbf{A}^{-1}\mathbf{b}$ die Approximation $\mathbf{x} \approx \mathbf{A}^{-1}\mathbf{b}$ folgt. Im Folgenden werden diejenigen Ansätze beschrieben, welche in Kapitel 9 kombiniert und verglichen werden. Da die Methoden nicht unbedingt für Parallelrechner konzipiert sind, wird das verteilte Gesamtsystem (8.1) dessen Daten auf die verschiedenen PEs verteilt sind, in einen Satz kleinerer, lokaler Gleichungssysteme

$$\mathbf{A}^k \mathbf{x}^k = \mathbf{b}^k, \quad k = 1, \dots, p \quad (8.2)$$

zerlegt, die von den p PEs sequentiell gelöst werden. Lösen bedeutet in diesem Kapitel, dass \mathbf{x}^k die Lösung der Gleichung (8.2) approximiert. Welche Form die lokalen Gleichungssysteme annehmen, hängt von der parallelen Vorkonditionierungsmethode ab.

8.1. Präkonditionierungsbausteine

8.1.1. Einfaches Skalieren

Das einfache Skalieren ist eine simple Präkonditionierungsmethode, die zusätzlich zu aufwendigeren Verfahren durchgeführt werden kann. Das globale Gleichungssystem wird mit einer regulären Diagonalmatrix $\mathbf{D} = \text{diag}\{d_{11}, d_{22}, \dots, d_{nn}\}$ multipliziert:

$$\mathbf{DAx} = \mathbf{Db}. \quad (8.3)$$

8. Präkonditionierungsmodelle

Es existieren verschiedene Ansätze, wie die Einträge d_{ii} , $i = 1, \dots, n$ definiert werden können. Als einfache, lokale Präkonditionierungsmethode eignet sich die Zeilenskalierung bezüglich der Betragssummennorm mit

$$d_{ii} = \left(\sum_{j=1}^n |a_{ij}| \right)^{-1} \quad (8.4)$$

sehr gut, da die Matrix zeilenweise verteilt ist. Somit kann jede PE für sich die entsprechenden Einträge der Diagonalmatrix \mathbf{D} erzeugen, ohne dass Daten kommuniziert werden müssen. Mit dem einfachen Skalieren wird das Risiko verringert, dass sehr kleine Zahlen mit sehr großen Zahlen addiert werden, was sich auf die numerische Genauigkeit auswirken kann [25]. Zudem ist die Wahl (8.4) im Rahmen der Skalierungen optimal [42]. Jedoch beschreibt die Diagonalmatrix \mathbf{D} nur eine sehr schlechte Approximation von \mathbf{A}^{-1} , so dass die Verwendung von weiteren Präkonditionierungsmethoden ratsam ist. Aber da sowohl die Datensynchronisation beim Aufsetzen, als auch eine zusätzliche Rechenlast beim wiederholten Anwenden fehlen, wird die Zeilenskalierung im Folgenden immer genutzt, auch wenn auf die Notation der Diagonalmatrix verzichtet wird.

8.1.2. Unvollständige LU-Zerlegung

Ein Verfahren, welches aus einem direkten Lösungsansatz entstanden ist, stellt die unvollständige LU-Zerlegung dar. Die Abkürzung ILU der englischen Bezeichnung Incomplete LU factorization ist für diese Methode auch im Deutschen üblich. Wie der Name impliziert, ist dieses Verfahren eine Abwandlung der (vollständigen) LU-Zerlegung, welche einen direkten Löser begründet, der in geeigneten Fällen sehr effektiv ist. In seiner vollständigen Durchführung wird eine allgemeine, reguläre, quadratische Matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ mittels Gauß-Elimination in eine linke, untere Dreiecksmatrix \mathbf{L} und eine rechte, obere Dreiecksmatrix \mathbf{U} zerlegt, so dass

$$\mathbf{LU} = \mathbf{A} \quad (8.5)$$

mit $\mathbf{L}, \mathbf{U} \in \mathbb{R}^{n \times n}$ gilt. Da Dreiecksmatrizen sehr einfach zu invertieren sind, kann das Gleichungssystem $\mathbf{LU}\mathbf{x} = \mathbf{b}$ mit dem Algorithmus

$$\tilde{\mathbf{b}} = \mathbf{L}^{-1}\mathbf{b} \quad (8.6)$$

$$\mathbf{x} = \mathbf{U}^{-1}\tilde{\mathbf{b}} \quad (8.7)$$

gelöst werden, wobei $\tilde{\mathbf{b}}$ in (8.6) durch Vorwärtselimination bestimmt wird, während eine Rückwärtselimination der Gleichung (8.7) die Berechnung von \mathbf{x} vollendet. Leider ist diese Herangehensweise keine alternative Lösungsstrategie für das gegebene Gleichungssystem, da die entsprechenden Dreiecksmatrizen \mathbf{L} und \mathbf{U} einer dünnbesetzten Matrix \mathbf{A} im Allgemeinen nicht gleichfalls dünnbesetzt sind. Da bei höherer Auflösung des Rechengitters die Größe der vollbesetzten Matrix quadratisch ansteigt, kann die anwachsende Speicheranforderung schnell nicht mehr befriedigt werden. Aus

dieser Problematik entspringt die Idee, die Grundstruktur des Verfahrens zur Präkonditionierung zu nutzen, wobei die Dreiecksmatrizen $\tilde{\mathbf{L}}$ und $\tilde{\mathbf{U}}$ erzwungenermaßen dünnbesetzt sind, jedoch die Matrix \mathbf{A} approximieren:

$$\tilde{\mathbf{L}}\tilde{\mathbf{U}} \approx \mathbf{A}. \quad (8.8)$$

Die Berechnung der Faktorisierung geschieht mittels Gauß-Elimination zeilenweise, siehe Abbildung 8.1. Somit kann der Vektor \mathbf{x}_i mit der Vorwärts- und Rückwärtselimination aus (8.6) und (8.7), angenähert werden. Trotz Verzicht der Tilde in der Notation der Dreiecksmatrizen wird im weiteren Verlauf nur die Approximation $\mathbf{L}\mathbf{U} \approx \mathbf{A}$ gemeint sein.

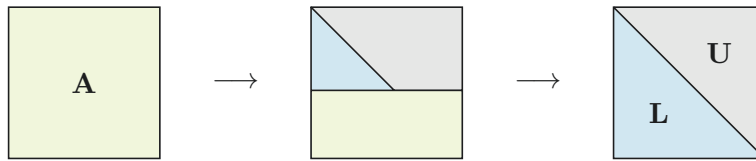


Abbildung 8.1.: Mittels Gauß-Elimination in (un-)vollständiger Version, siehe Algorithmus 8.1, wird die Matrix \mathbf{A} Zeile für Zeile faktorisiert.

Im Hinblick auf die Parallelisierung ist jegliche Art von LU-Zerlegung denkbar ungünstig. Bei einer verteilten Matrix können die einzelnen PEs im Falle einer Vorwärtsbeziehungsweise Rückwärtselimination nur nacheinander und nicht nebenläufig ihren Teil zur Lösung des Systems beitragen. Dennoch wird dieser Abschnitt dieser Methode gewidmet, da sie bei einem Satz lokaler Gleichungssysteme, siehe Gleichung (8.2), von jeder PE sequentiell eingesetzt werden kann.

Die Art und Weise, welche Einträge in \mathbf{L} und \mathbf{U} unterdrückt oder vernachlässigt werden, um die dünnbesetzte Struktur zu gewähren, ist je nach ILU-Variante unterschiedlich. Um die Notation aus [19] zu übernehmen, bestimmt die Tupelmengens S , an welchen Stellen Einträge ungleich Null erlaubt sind:

$$l_{ij} = \begin{cases} l_{ij} & : i \leq j, (i, j) \in S, \\ 0 & : \text{sonst,} \end{cases} \quad u_{ij} = \begin{cases} u_{ij} & : i \geq j, (i, j) \in S, \\ 0 & : \text{sonst.} \end{cases} \quad (8.9)$$

Die Menge S beschreibt entspricht der Indexmenge S_{L+U} .

Die sehr simple Vorschrift von ILU(0) besagt, dass ein Nichtnulleintrag nur dort zugelassen wird, wo der entsprechende Eintrag in der Matrix \mathbf{A} ungleich Null ist:

$$S = S_A = \{(i, j) \quad : \quad a_{ij} \neq 0\}. \quad (8.10)$$

Das Füllmuster von $\mathbf{L} + \mathbf{U}$ gleicht demnach dem von \mathbf{A} , siehe Abbildung 8.2. Bei der Faktorisierung der Matrix kann also allein mit einer `if`-Abfrage geklärt werden, welche Werte berechnet und gespeichert werden müssen. Die Einfachheit der Vorschrift führt jedoch zu einer vergleichsweise schlechten Approximation, denn es ist sehr wahrscheinlich, dass wichtige Werte unterdrückt werden. Wie dieser Ansatz zu seinem Namen kam, erschließt sich im nächsten Absatz.

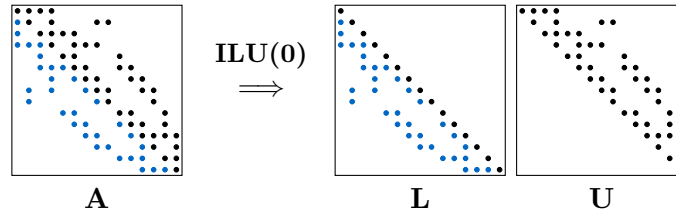


Abbildung 8.2.: Matrixstrukturen bei der ILU(0)-Faktorisierung.

Eine effizientere Variante bietet die unvollständige LU-Zerlegung mit dem Level k , kurz ILU(k). Dieser Ansatz verfolgt eine sinnvolle Regelung, die vorgibt, welche zusätzlichen Einträge ungleich Null erlaubt werden, während die Gesamtanzahl dieser Einträge beschränkt bleibt. Dabei wird jedem Indexpaar (i, j) ein Startlevel α_{ij} zugeordnet:

$$\alpha_{ij} = \begin{cases} 0 & : (i, j) \in S_A, \\ \infty & : \text{sonst.} \end{cases} \quad (8.11)$$

Im Verlauf der Faktorisierung, wird bei der Berechnung von l_{ij} beziehungsweise u_{ij} das Level α_{ij} neu bestimmt. Ausgehend von der Gauß-Elimination wird in der innersten Schleife zusätzlich zu $a_{ij} := a_{ij} - a_{ik}a_{kj}$ das zugehörige Indexlevel α_{ij} aktualisiert und auf

$$\alpha_{ij} := \min(\alpha_{ij}, \alpha_{ik} + \alpha_{kj} + 1) \quad (8.12)$$

gesetzt, vergleiche Algorithmus 8.1 für ILU(k). Alle Einträge, deren Level k überschreitet, werden unterdrückt. Die Indexmenge beschränkt sich auf

$$S = \{(i, j) : \alpha_{ij} \leq k\}. \quad (8.13)$$

Da der Levelindex weder anwachsen, noch von ∞ auf 0 fallen kann, führt $k = 0$ zu Definition (8.10), was die Namensgebung dieser Variante erklärt.

Statt die Menge S über die Indexmenge S_A zu definieren, wird bei der Ausführung von ILUT (ILU Threshold) mit Hilfe zweier Kriterien der Wert der Einträge überprüft,

$$S_\tau = \{(i, j) : |a_{ij}| > \tau\}, \quad (8.14)$$

$$S_f = \{(i, j) : |\{k : |a_{ik}| > |a_{ij}|\}| < n_f\}, \quad (8.15)$$

und es gilt

$$S = S_\tau \cap S_f. \quad (8.16)$$

Mit (8.14) werden die Werte vernachlässigt, die betragsmäßig eine gewählte Grenze τ unterschreiten. Unabhängig von der Position werden die kleinen Werte unterdrückt. Zusätzlich wird der Füllparameter n_f berücksichtigt, der die maximale Anzahl der Nichtnulleinträge pro Zeile beschreibt. Erfüllen mehr als n_f Werte die erste Bedingung, so werden die betraglich kleineren ebenfalls unterdrückt. Dies gewährt auch bei schlecht gewähltem τ eine Beschränkung der Speicheranforderungen. Für die Loslösung von der Positionsabhängigkeit müssen jedoch bei der Faktorisierung zeilenweise alle Einträge berechnet, sortiert und dann überprüft werden. Dies ist deutlich mehr Aufwand, als eine einfache `if`-Abfrage zu Beginn. Dafür wird jedoch auch eine sinnvollere Zerlegung geboten.

Algorithmus 8.1: Die Gauß-Elimination zur vollständigen LU-Zerlegung und die Modifikationen zu den Varianten ILU(0), ILU(K) und ILUT im Pseudo-Code.

LU	ILU(0)
<pre> for₁ (i = 2, i ≤ n, i++) for₂ (k = 1, k ≤ i - 1, k++) a_{ik} := a_{ik}/a_{kk} for₃ (j = k + 1, j ≤ n, j++) a_{ij} := a_{ij} - a_{ik}a_{kj} </pre>	<pre> for₁ (i = 2, i ≤ n, i++) for₂ (k = 1, k ≤ i - 1, k++) a_{ik} := a_{ik}/a_{kk} for₃ (j = k + 1, j ≤ n, j++) if ((i, j) ∈ S_A) a_{ij} := a_{ij} - a_{ik}a_{kj} </pre>
ILU(K)	ILUT
<pre> α_{ij} := (8.11) for₁ (i = 2, i ≤ n, i++) for₂ (k = 1, k ≤ i - 1, k++) a_{ik} := a_{ik}/a_{kk} for₃ (j = k + 1, j ≤ n, j++) if (α_{ij} ≤ K) a_{ij} := a_{ij} - a_{ik}a_{kj} α_{ij} := min(α_{ij}, α_{ik}+α_{kj}+1) </pre>	<pre> for₁ (i = 2, i ≤ n, i++) for₂ (k = 1, k ≤ i - 1, k++) a_{ik} := a_{ik}/a_{kk} for₃ (j = k + 1, j ≤ n, j++) a_{ij} := a_{ij} - a_{ik}a_{kj} for₄ (j = 1, j ≤ n, j++) if ((i, j) ∉ S_τ ∩ S_f) a_{ij} := 0 </pre>

8.2. Vom globalen zum lokalen System

In diesem Abschnitt werden zwei Verfahren vorgestellt, welche ausgehend von dem globalen Gleichungssystem (8.1) einen Satz lokaler Systeme (8.2) bilden. Nach der Definition dieser kleineren Systeme werden (falls notwendig) fehlende Daten synchronisiert und jede PE löst mittels unvollständiger LU-Faktorisierung das ihr zugewiesene System. Dies geschieht in jedem Iterationsschritt des Krylov-Unterraumverfahrens. Da die Präkonditionierung das Lösungsverfahren beschleunigen soll, muss dabei beachtet werden, welche zusätzlichen Rechenlasten erzeugt werden, sowohl beim Aufsetzen als auch bei der Anwendung in jedem Iterationsschritt. Der Zusammenhang, dass eine bessere Präkonditionierung zu schnellerer Konvergenz und somit weniger Iterationsschritten führt, ist dabei nicht zu vernachlässigen.

Im Vergleich verschiedener Vorgehensweisen muss beachtet werden, inwiefern der Satz lokaler Gleichungssysteme mit der Gebietszerlegung zusammenhängt, beziehungsweise, auf welche Teilgebiete das globale System verteilt wird. Bei genauerer Betrachtung der globalen verteilten Matrix \mathbf{A} fällt auf, dass nur wenige Blöcke außerhalb der Blockdiagonalen nichttrivial sind, wie in Abbildung 8.3 schematisch dargestellt. Nur Blöcke benachbarter Partitionen können Nichtnulleinträge enthalten. In diesen Blöcken sind nur die Einträge ungleich Null, die Interaktionen zwischen den Partitionen beschreiben, was nur auf den Grenzknoten geschieht.

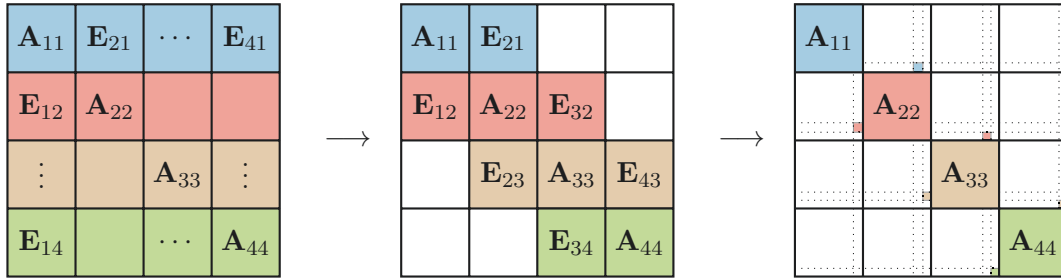


Abbildung 8.3.: Blockbetrachtung der dünnbesetzten Matrix \mathbf{A} . Die Einfärbung signalisiert, welche PE die Einträge lokal gespeichert hat.

Für die Reduzierung des Gesamtsystems auf ein Teilgebiet, das eine Menge \mathcal{M} mit m Stützknoten umfasst, wird ein Restriktionsoperator definiert. Dieser kann als dünnbesetzte Matrix $\mathbf{R}_{\mathcal{M}} \in \mathbb{R}^{m \times n}$ dargestellt werden, wobei die Einträge bezogen auf die Indexmenge

$$S_{R,\mathcal{M}} = \{(i, \pi(i)) \quad : \quad i = 1, \dots, m \wedge n_{\pi(i)} \in \mathcal{M}\} \quad (8.17)$$

auf 1 gesetzt werden. Dabei beschreibt die Funktion π die Abbildung vom lokalen zum globalen Index. In der Form

$$\mathbf{R}_{\mathcal{M}} \mathbf{A} \mathbf{R}_{\mathcal{M}}^T = \mathbf{A}|_{\mathcal{M}} \in \mathbb{R}^{m \times m} \quad (8.18)$$

kann nun die Matrix \mathbf{A} entsprechend der Wahl der Knotenmenge \mathcal{M} auf eine Teilmatrix reduziert werden, siehe Abbildung 8.4.

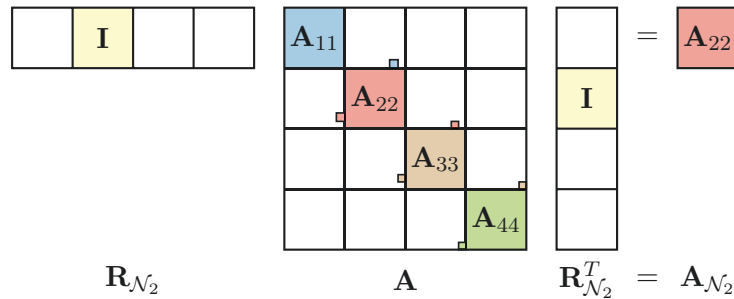


Abbildung 8.4.: Restriktion der Matrix \mathbf{A} auf den Diagonalblock \mathbf{A}_{22} . Die Blockmatrix \mathbf{I} beschreibt die Einheitsmatrix.

8.2.1. Block-Jacobi Verfahren

Das von Grund auf einfachste Verfahren liefert der Block-Jacobi (BJ) Ansatz. Der Name dieser Methode rührt vom ursprünglichen, (ungeblockten) Jacobi Verfahren her, für dessen Präkonditionierungsmatrix nur die Werte auf der Diagonalen verwendet werden. Bei einer Zerlegung von $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$ wird dabei $\mathbf{K}^{-1} = \mathbf{D}^{-1}$ gesetzt, wobei

\mathbf{D} eine Diagonalmatrix, \mathbf{L} eine linke untere und \mathbf{U} eine rechte obere Dreiecksmatrix beschreibt. Die Block-Jacobi Variante erzielt bessere Ergebnisse, da viele Einträge von \mathbf{A} beachtet werden, welche im Originalverfahren rigoros vernachlässigt wurden, was allerdings für Einträge außerhalb der Diagonalblöcke immer noch geschieht. In Abbildung 8.5 sind links in der globalen Matrix die Blöcke farblich markiert, die von den einzelnen PEs bei der Prädiktionierung mit einbezogen werden. Rechts sind die Gleichungssysteme skizziert, die lokal mit Hilfe einer unvollständigen LU-Zerlegung gelöst werden. Es werden also ausschließlich die Diagonalblöcke \mathbf{A}_{kk} , $k = 1, \dots, p$ betrachtet.

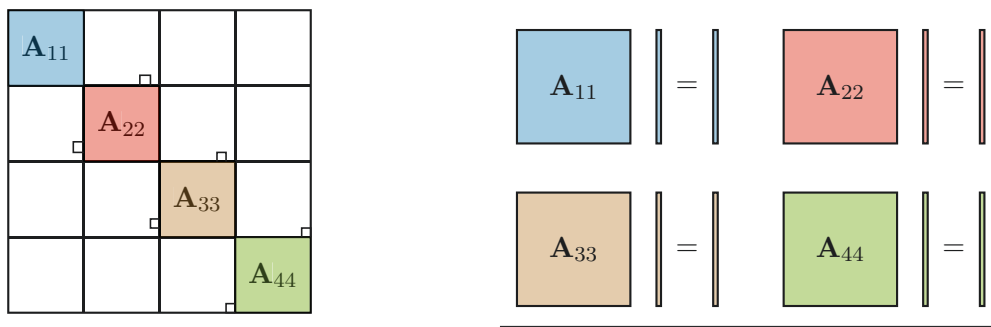


Abbildung 8.5.: Beim Block-Jacobi-Verfahren werden nur die lokalen Diagonalblöcke betrachtet.

Sequentiell approximiert nun jede PE mittels gewählter ILU-Variante die Lösung des lokalen Systems

$$\mathbf{A}_{kk} \mathbf{x}_k = \mathbf{b}_k. \quad (8.19)$$

Da jede PE nur den von ihr kontrollierten Teil \mathbf{x}_k des approximativen Lösungsvektors berechnet hat, müssen die Werte im Anschluss nicht synchronisiert werden. Mit Hinblick auf die Gebietszerlegung entspricht $\mathbf{A}_{kk} = \mathbf{R}_{\mathcal{N}_k} \mathbf{A} \mathbf{R}_{\mathcal{N}_k}^T$. In diesem Fall wird unter Ausnahme aller Grenzelemente ein Satz disjunkter Rechengebiete betrachtet, welche ausschließlich die jeweilige Knotenmenge \mathcal{N}_k umfassen. Des Weiteren zeigt der Vergleich mit Gleichung (6.5), dass in (8.19) der Summenterm komplett vernachlässigt wird. Beide Betrachtungsweisen, lassen darauf schließen, dass die Approximation der Inversen \mathbf{A}^{-1} von minderer Qualität sein kann, da der Einfluss der Gleichungsterme über Gebietsgrenzen hinweg im Allgemeinen nicht geringer ist, als innerhalb des Teilgebiets.

Erwartungsgemäß erzielt diese mangelhafte Approximation für gewöhnlich keine hohe Konvergenzrate, da die Beschleunigung des Lösungsverfahrens bezogen auf die Anzahl der Iterationsschritte gering ist. Dies wird auch in Abschnitt 9.2.3 beim Vergleich mit den anderen Prädiktionierungsmethoden bestätigt. Als wahrer Vorteil muss jedoch genannt werden, dass jegliches Vermeiden von Datensynchronisation während Konfiguration und Anwendung innerhalb der Prädiktionierungsroutine Verklemmungen vermeidet, da die verschiedenen PEs nicht aufeinander warten müssen. Somit ist pro Iterationsschritt des Krylov-Unterraumverfahrens die Prädiktionierung an sich nicht teuer.

8.2.2. Restricted Additive Schwarz Methode

Eine deutlich bessere Annäherung an die Inverse von \mathbf{A} liefert die RAS Methode, welche als Variante der Additive Schwarz Methode in [11] präsentiert wurde. Pro PE wird hier das Teilgebiet betrachtet, das von der Knotenmenge \mathcal{N}_k^1 aufgespannt wird. Dies führt hinsichtlich der zugehörigen Matrixeinträge zu einem überlappenden System. Dementsprechend werden die Restriktionsoperatoren $\mathbf{R}_{\mathcal{N}_k^1}$ konstruiert und die einzelnen PEs approximieren mit Hilfe einer ILU-Version die Lösung von

$$\mathbf{A}_k^{\text{ext}} \mathbf{x}_k^{\text{ext}} = \mathbf{b}_k^{\text{ext}}. \quad (8.20)$$

Dies ist möglich, da $\mathbf{A}_k^{\text{ext}} = \mathbf{R}_{\mathcal{N}_k^1} \mathbf{A} \mathbf{R}_{\mathcal{N}_k^1}^T \in \mathbb{R}^{m_k^e \times m_k^e}$ quadratisch, $\mathbf{x}_k^{\text{ext}} \in \mathbb{R}^{m_k^e}$ und $\mathbf{b}_k^{\text{ext}} = \mathbf{R}_{\mathcal{N}_k^1} \mathbf{b} \in \mathbb{R}^{m_k^e}$.

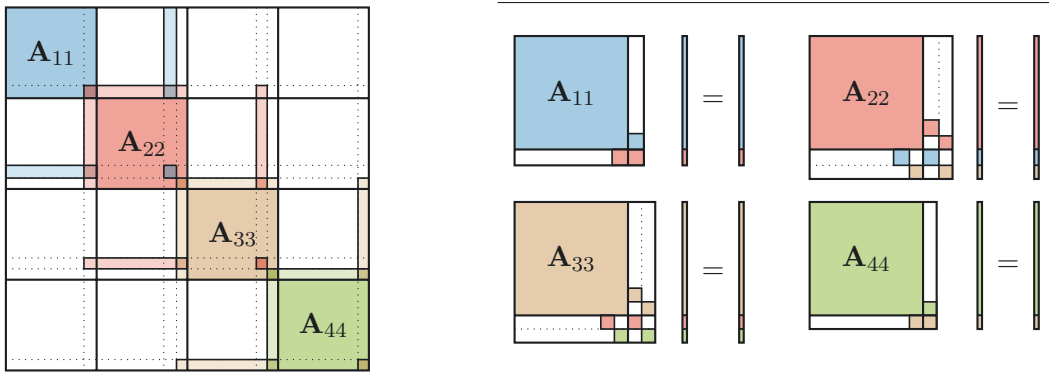


Abbildung 8.6.: Bei der RAS-Präkonditionierung überlappen sich die lokalen Teilmatrizen. Die Einfärbung zeigt an, welche Einträge der globalen Matrix sich in den Teilmatrizen $\mathbf{A}_k^{\text{ext}}$ wiederfinden.

In Abbildung 8.6 sind wieder die lokalen Gleichungssysteme und deren Äquivalent in der globalen Matrix dargestellt. Der Überlapp der einzelnen Systeme ist dabei klar erkennbar. Die Summenterme aus (6.5), welche bei der Block-Jacobi Methode vernachlässigt werden, sind in diesem Ansatz in der erweiterten Matrix $\mathbf{A}_k^{\text{ext}}$ mit einbezogen. So werden in diesem Fall beim Aufbau der Gleichungssysteme keinerlei Einträge der Matrix übergangen. Dies kostet allerdings zusätzliche Ressourcen an Speicherplatz und Rechenzeit beim Aufbau des Präkonditionierers, da die Einträge $\{a_{ij} : n_i \in \mathcal{N}_k^1 \setminus \mathcal{N}_k \wedge n_j \in \mathcal{N}_k^1\}$ auf externen Speichern liegen und Replikation erfordern. Bei der Anwendung muss zudem die Erweiterung der rechten Seite $\mathbf{b}_k^{\text{ext}}$ synchronisiert werden. Nach der Bestimmung des Vektors $\mathbf{x}_k^{\text{ext}}$ gibt es bezüglich der Grenzknoten unterschiedliche Lösungen auf den verschiedenen PEs. Im klassischen Additive Schwarz Verfahren werden den Überlapp betreffende Werte deshalb aufsummiert und gemittelt. In [11] wurde jedoch gezeigt, dass das Verfahren schneller konvergiert, wenn auf diesen Abgleich verzichtet wird und ausschließlich die Werte bezüglich \mathcal{N}_k je Teilvektor $\mathbf{x}_k^{\text{ext}}$ betrachtet werden. Statt $\mathbf{x}_k^{\text{ext}}$ wird wie beim BJ-Verfahren der globale Vektor \mathbf{x} aus den Teilvektoren $\mathbf{x}_k = \mathbf{x}_k^{\text{ext}}|_{\mathcal{N}_k}$ zusammengesetzt.

Da die Knotenmengen \mathcal{N}_k , $k = 1, \dots, p$ disjunkt sind, siehe Gleichung (6.2), und die zu verwendenden Werte schon im richtigen Speicher liegen, wird dazu kein Datenaustausch und keine Arithmetik benotigt. So wird mit weniger Aufwand eine bessere Konvergenzrate erreicht. Die benotigte Kommunikation ist auf den Datenaustausch fur die Konfiguration und die Synchronisation der rechten Seite bei der Anwendung beschrankt. Im Vergleich zum BJ-Verfahren wird dennoch mehr Datensynchronisation und zudem mehr Rechenoperationen getatigt, da die Dimension des Gleichungssystems (8.20) um $m_k^e - m_k$ groer ist, als des Teilsystems (8.19). Doch spiegelt sich die bessere Approximation von \mathbf{A}^{-1} in der geringeren Anzahl von Iterationsschritten des Krylov-Unterraumverfahrens wieder, wie in Abschnitt 9.2.3 gezeigt wird.

8.3. Verschachtelte Prakonditionierungsmethoden

In diesem Abschnitt wird nicht nur eine, sondern werden zwei Prakonditionierungsmethoden vorgestellt. Beide Techniken entspringen demselben Ansatz: basierend auf dem Schurkomplement wird zur Prakonditionierung ein viel kleineres globales Gleichungssystem berechnet, das dann selbst ebenfalls vorkonditioniert wird. Bei genauerem Hinsehen erschliet sich, dass die zwei erwahnten Methoden im Prinzip durch Anwendung von BJ beziehungsweise dem RAS auf das kleinere Gleichungssystem geschaffen werden.

8.3.1. Das Schurkomplement

In Abschnitt 6.2.1 uber die Gebietszerlegung wurde eine Klassifikation der lokalen Gitterknoten in innere Knoten und Grenzknoten vorgenommen und die lokale Sortierung entsprechend angepasst. Diese lokale Gruppierung in innere und Grenzknoten stellt hier den Dreh- und Angelpunkt dar. Ausgehend von den Gleichungssystemen (6.5) entsteht auf diese Weise, bei einer entsprechenden Blockbildung in Matrizen und Vektoren, die Formulierung

$$\underbrace{\begin{pmatrix} \mathbf{B}_k & \mathbf{F}_k \\ \mathbf{E}_k & \mathbf{C}_k \end{pmatrix}}_{\mathbf{A}_{kk}} \underbrace{\begin{pmatrix} \mathbf{u}_k \\ \mathbf{y}_k \end{pmatrix}}_{\mathbf{x}_k} + \sum_{j \neq k} \underbrace{\begin{pmatrix} 0 & 0 \\ 0 & \bar{\mathbf{E}}_{kj} \end{pmatrix}}_{\mathbf{E}_{kj}} \underbrace{\begin{pmatrix} \mathbf{u}_j \\ \mathbf{y}_j \end{pmatrix}}_{\mathbf{x}_j} = \underbrace{\begin{pmatrix} \mathbf{f}_k \\ \mathbf{g}_k \end{pmatrix}}_{\mathbf{b}_k}, \quad (8.21)$$

wobei die Darstellung jener aus [56] angelehnt ist. Die Beschrankung von \mathbf{x}_k auf die Menge der inneren Knoten wird demnach mit \mathbf{u}_k beschrieben, wahrend $\mathbf{x}_k|_{\mathcal{Q}_k} = \mathbf{y}_k$ die Komponenten bezuglich der Grenzknoten darstellt. Zur Berechnung der Losungskomponenten innerer Knoten kann dieser Teil des Gleichungssystems zu

$$\mathbf{u}_k = \mathbf{B}_k^{-1}(\mathbf{f}_k - \mathbf{F}_k \mathbf{y}_k), \quad (8.22)$$

umgeformt werden. Eingesetzt in den unteren Teil des Systems, folgt

$$\mathbf{S}_k \mathbf{y}_k + \sum_{j \neq k} \bar{\mathbf{E}}_{kj} \mathbf{y}_j = \tilde{\mathbf{g}}_k, \quad (8.23)$$

8. Präkonditionierungsmodelle

wobei das lokale Schurelement \mathbf{S}_k und die entsprechende rechte Seite $\tilde{\mathbf{g}}_k$ die Gestalt

$$\mathbf{S}_k := \mathbf{C}_k - \mathbf{E}_k \mathbf{B}_k^{-1} \mathbf{F}_k \quad \text{und} \quad \tilde{\mathbf{g}}_k := \mathbf{g}_k - \mathbf{E}_k \mathbf{B}_k^{-1} \mathbf{f}_k, \quad (8.24)$$

haben. In Abbildung 8.7 wird bezogen auf eine Partition die Reduktion des Gleichungssystems auf die Grenzknoten gezeigt, wobei auf die Dimensionsunterschiede der beiden Systeme zu achten ist.

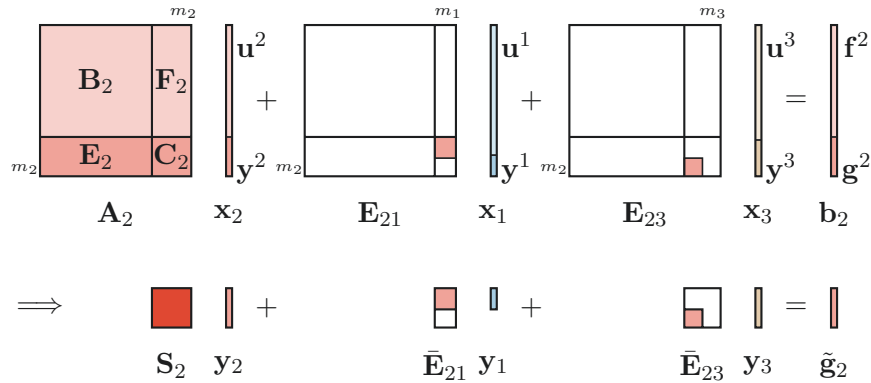


Abbildung 8.7.: Mit Hilfe des Schurkomplements kann das Gleichungssystem beschränkt auf die Grenzknoten dargestellt werden.

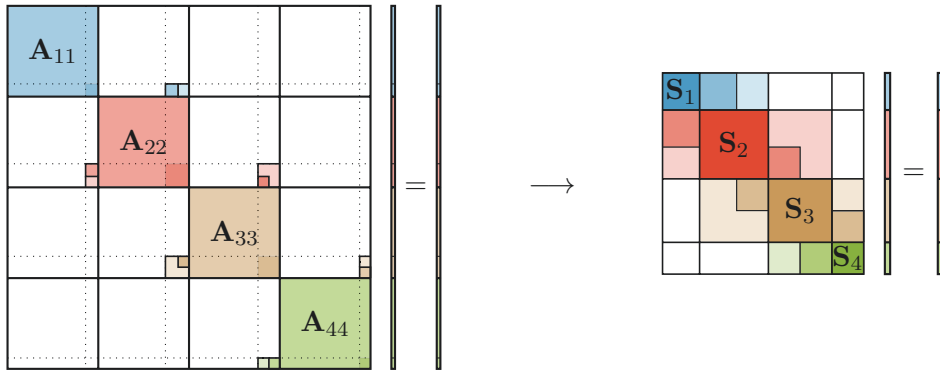


Abbildung 8.8.: Das lineare Gleichungssystem $\mathbf{A}\mathbf{x} = \mathbf{b}$ wird übergeführt zu $\mathbf{S}\mathbf{y} = \tilde{\mathbf{g}}$, um die Lösung bezogen auf die Grenzknoten zu bestimmen. Die übrigen Komponenten von \mathbf{x} können daraufhin explizit berechnet werden.

Bei exakter Berechnung von \mathbf{y}_k in (8.23), kann \mathbf{u}_k mit (8.22) explizit bestimmt werden. Hier wird dieses Verfahren ausschließlich zur Vorkonditionierung verwendet, so dass an verschiedenen Stellen approximiert wird. Der Satz lokaler Gleichungssysteme, definiert

in (8.23), kann zu einem globalen System zusammengefasst werden,

$$\underbrace{\begin{pmatrix} \mathbf{S}_1 & \bar{\mathbf{E}}_{12} & \cdots & \bar{\mathbf{E}}_{1p} \\ \bar{\mathbf{E}}_{21} & \mathbf{S}_2 & \cdots & \bar{\mathbf{E}}_{2p} \\ & & \ddots & \\ \bar{\mathbf{E}}_{p1} & \cdots & \bar{\mathbf{E}}_{p,p-1} & \mathbf{S}_p \end{pmatrix}}_{\mathbf{S}} \underbrace{\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_p \end{pmatrix}}_{\mathbf{y}} = \underbrace{\begin{pmatrix} \tilde{\mathbf{g}}_1 \\ \tilde{\mathbf{g}}_2 \\ \vdots \\ \tilde{\mathbf{g}}_p \end{pmatrix}}_{\tilde{\mathbf{g}}} \quad (8.25)$$

wie in Abbildung 8.8 in einem einfachen Beispiel dargestellt. Mit diesem Ansatz wird das Gleichungssystem (8.1) mit n Unbekannten zu einem Gleichungssystem mit $n_S = n - \sum m_k^i$ Unbekannten reduziert, das auf die Knotenmenge $\bigcup \mathcal{Q}_k$ beschränkt ist. Die Diagonalblöcke der globalen Schurmatrix \mathbf{S} sind durch die lokalen Schurkomplemente \mathbf{S}_k , für $1 \leq k \leq p$ vertreten. Die Offdiagonale bilden die auf Grenzknoten beschränkte Blöcke $\bar{\mathbf{E}}_{kj} = \mathbf{R}_{\mathcal{Q}_{kj}} \mathbf{E}_{kj} \mathbf{R}_{\mathcal{Q}_{kj}}^T$ der Dimension $\dim(\bar{\mathbf{E}}_{kj}) = m_k^g \times m_j^g$. Leider ist die exakte Bestimmung des lokalen Schurkomplements nach (8.24) mit den Matrix-Matrix-Multiplikationen sehr rechenintensiv. Doch kann wie in [56], auch mit Hilfe des Gauß-Eliminationsverfahrens eine Überführung zum lokalen Schursystem bewerkstelligt werden, indem das Verfahren nach m_k^i Schritten abgebrochen wird. Somit wird eine Zerlegung $\mathbf{A}_{kk} = \mathbf{L}_k \mathbf{U}_k$ in Blockdreiecksmatrizen im Sinne von

$$\mathbf{L}_k := \begin{pmatrix} \mathbf{L}_{\mathbf{B}_k} & 0 \\ \mathbf{E}_k \mathbf{U}_{\mathbf{B}_k}^{-1} & \mathbf{I} \end{pmatrix} \quad \text{und} \quad \mathbf{U}_k := \begin{pmatrix} \mathbf{U}_{\mathbf{B}_k} & \mathbf{L}_{\mathbf{B}_k}^{-1} \mathbf{F}_k \\ 0 & \mathbf{S}_k \end{pmatrix}, \quad (8.26)$$

vollbracht, wie in Abbildung 8.9 illustriert.

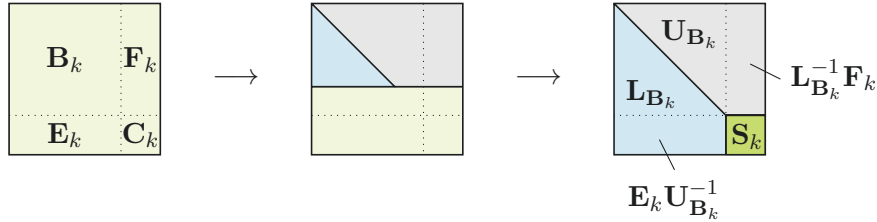


Abbildung 8.9.: Durch Abbruch des Gauß-Eliminationsverfahren können die Blockdreiecksmatrizen \mathbf{L}_k und \mathbf{U}_k gemäß (8.26) einfach bestimmt werden.

Nun wird via Vorwärtssubstitution $\tilde{\mathbf{b}}_k = \mathbf{L}_k^{-1} \mathbf{b}_k$ berechnet, wobei $\tilde{\mathbf{b}}_k|_{\mathcal{Q}_k} = \tilde{\mathbf{g}}_k$ aus (8.24) entspricht. Mittels Rücksubstitution $\mathbf{x}_k = \mathbf{U}_k^{-1} \tilde{\mathbf{b}}_k$ wird erst die Lösung \mathbf{y}_k auf den Grenzknoten ermittelt und anschließend die Komponenten bezogen auf die inneren Knoten bestimmt, was gerade der Gleichung (8.22) entspricht. Diese Vorgehensweise folgt genau der Vorwärts- und Rückwärtssubstitution aus (8.6) - (8.7). Bei genauem Hinsehen fällt auf, dass streng genommen bei der Rücksubstitution die Werte auf den Grenzknoten mit

$$\mathbf{y}_k = \mathbf{S}_k^{-1} \tilde{\mathbf{g}}_k, \quad (8.27)$$

berechnet werden und die Interaktion über die Partitionsgrenzen hinweg vernachlässigt wird. Als Vorkonditionierung ist diese Vorgehensweise durchaus möglich, jedoch

8. Präkonditionierungsmodelle

liefert die Methode eine bessere Annäherung an die Lösung, wenn der Ablauf bezüglich Aufspaltung in innere und Grenzknoten beibehalten wird, wie in Algorithmus 8.2 aufgeschlüsselt, jedoch bei der Rücksubstitution statt (8.27) die Lösung der Gleichung (8.23) mit Hilfe von bekannten Präkonditionierungsmethoden und Lösungsverfahren angenähert wird.

Algorithmus 8.2: Schrittweise Vorwärts- und Rückwärtssubstitution, wobei Werte bezüglich innerer und Grenzknoten getrennt voneinander bestimmt werden.

	interne Knoten	Grenzknoten
1	$\tilde{\mathbf{f}}_k := \mathbf{L}_{\mathbf{B}_k}^{-1} \mathbf{f}_k$	
2		$\tilde{\mathbf{g}}_k := \mathbf{g}_k - \mathbf{E}_k \mathbf{U}_{\mathbf{B}_k}^{-1} \tilde{\mathbf{f}}_k$
3		$\mathbf{y}_k := \text{solve}(\mathbf{S}_k \mathbf{y}_k + \sum \bar{\mathbf{E}}_{kj} \mathbf{y}_j = \tilde{\mathbf{g}}_k)$
4	$\mathbf{u}_k := \mathbf{U}_{\mathbf{B}_k}^{-1} (\tilde{\mathbf{f}}_k - \mathbf{L}_{\mathbf{B}_k}^{-1} \mathbf{F}_k \mathbf{y}_k)$	

8.3.2. Vorkonditionierung der Schursystems

Aus oben erwähnten Gründen kommt eine vollständige LU-Zerlegung der Matrix \mathbf{A} nicht in Frage. Das Schursystem wird hier nicht als Lösungs- sondern als Präkonditionierungsverfahren genutzt. Die Faktorisierung (8.26) wird deshalb mit einer der ILU-Varianten aus Abschnitt 8.1.2 vollführt, was zu einem approximierten Schurkomplement \mathbf{S}_k und einem angenähertem Schursystem (8.25) führt. Dabei wird eine Verschachtelung von Präkonditionierungsmethoden angewandt: Das dem Gleichungssystem (8.1) entsprechende Schursystem (8.25) wird approximiert und dessen Lösung wiederum nur angenähert. Die approximative Lösung auf den inneren Knoten wird anschließend mit (8.22) bestimmt. Erfreulicherweise muss für die unvollständige LU-Zerlegung von \mathbf{S}_k das lokale Schurkomplement nicht explizit bekannt sein, so dass auf die Matrix-Matrix-Multiplikationen aus Darstellung (8.24) verzichtet werden kann. Die Faktorisierung $\mathbf{S}_k \approx \mathbf{L}_{\mathbf{S}_k} \mathbf{U}_{\mathbf{S}_k}$ fällt bei der Faktorisierung (ohne Abbruch) der lokalen Matrix \mathbf{A}_{kk} als Nebenprodukt ab [56], wie folgende Darstellung der Zerlegung zeigt:

$$\begin{pmatrix} \mathbf{B}_k & \mathbf{F}_k \\ \mathbf{E}_k & \mathbf{C}_k \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{\mathbf{B}_k} & 0 \\ \mathbf{E}_k \mathbf{U}_{\mathbf{B}_k}^{-1} & \mathbf{L}_{\mathbf{S}_k} \end{pmatrix} \begin{pmatrix} \mathbf{U}_{\mathbf{B}_k} & \mathbf{L}_{\mathbf{B}_k}^{-1} \mathbf{F}_k \\ 0 & \mathbf{U}_{\mathbf{S}_k} \end{pmatrix}. \quad (8.28)$$

Die erste Möglichkeit besteht darin, das BJ-Verfahren auf die globale Schurgleichung (8.25) anzuwenden, um das lokale Gleichungssystem (8.27) zu erhalten. Dazu muss die Matrix \mathbf{S} nicht einmal explizit bekannt sein. Jedoch wird in diesem Fall nur das BJ-Verfahren aus Abschnitt 8.2.1 schrittweise ausgeführt. Um eine bessere Approximation von \mathbf{y}^k zu erzielen, wird statt (8.27) nun mit dem linksseitig vorkonditioniertem Krylov-Unterraumverfahren GMRES aus Abschnitt 7.5 das Gleichungssystem (8.23) iterativ gelöst. Als Präkonditionierer wird dabei $\mathbf{K}^{-1} = \mathbf{U}_{\mathbf{S}_k}^{-1} \mathbf{L}_{\mathbf{S}_k}^{-1}$ mit den Dreiecksmatrizen aus (8.28) verwendet [56].

Da es sich an dieser Stelle um eine Präkonditionierung der Präkonditionierung handelt und der innere Löser in jedem Iterationsschritt des Löser des Gleichungssystems

(7.1) aufgerufen wird, sind die Abbruchkriterien viel schwacher formuliert. Die ILU-Faktorisierung des lokalen Schurkomplements \mathbf{S}_k verursacht keine Datensynchronisation, da \mathbf{S}_k nur von Werten aus dem lokalen Diagonalblock \mathbf{A}_{kk} abhangt. Die Durchfuhrung des inneren GMRES-Verfahrens erzeugt allerdings pro Iterationsschritt bei der Konstruktion der Orthonormalisierung der Basisvektoren Datenkommunikation. Die Relation, dass das Erhohen der maximalen Iterationsschrittsanzahl innerhalb des inneren Losers, zwar eine genauere Approximation von \mathbf{y}^k liefert, jedoch seinen (nicht geringen) Preis hat, ist der Tatsache gegenuberzustellen, dass eine bessere Approximation innerhalb des Prakonditionierers die benotigte Anzahl an Iterationsschritten im externen Loser verringert.

Als Alternativmoglichkeit gibt es naturlich noch die Anwendung der Restricted Additive Schwarz Methode auf die Schurgleichung (8.25). Dieser Ansatz wird kurz SchurRAS genannt und in [37] besprochen. Hierbei wird kein innerer Losungsalgorithmus verwendet, sondern wie in Abschnitt 8.2.2 beschrieben, das erweiterte System $\mathbf{S}_{\text{ext}}^k \mathbf{y}_{\text{ext}}^k = \tilde{\mathbf{g}}_{\text{ext}}^k$ approximiert, indem $\mathbf{S}_{\text{ext}}^k = \mathbf{R}_{Q_k^1} \mathbf{S} \mathbf{R}_{Q_k^1}^T$ unvollstandig in Dreiecksmatrizen zerlegt wird. Im Gegensatz zu der Kombination Schur+GMRES ist hier das Aufstellen des Prakonditionierers aufwendiger, doch wird in der Anwendung kaum Datenkommunikation benotigt. Statt die globale Schurmatrix \mathbf{S} explizit aufzustellen, werden wie zuvor die Diagonalblocke \mathbf{A}_k der unvollstandigen LU-Zerlegung gema (8.28) unterworfen, wobei die Gau-Elimination bei m_k^i abgebrochen wird. Die nun bekannten (approximierten) lokalen Schurkomplemente \mathbf{S}_k , $1 \leq k \leq p$, welche die Diagonalblocke der globalen Schurmatrix \mathbf{S} bilden, werden zu $\mathbf{S}_k^{\text{ext}}$ erweitert, analog zur Erweiterung der Diagonalblocke der Matrix \mathbf{A} in Abschnitt 8.2.2. Da in der Implementierung nur Nichtnulleintrage synchronisiert werden und diese Eintrage in \mathbf{S} und \mathbf{A} bis auf eine Indexverschiebung von $-m_k^i$ identisch sind, konnen bei der Anwendung von RAS auf beide Matrizen fur die Erweiterung dieselben Funktionen verwendet werden.

9. Effiziente Berechnungsmethoden

9.1. Übersicht

Das Lösen des linearen Gleichungssystems (3.22) kostet mit Abstand die meiste Zeit bei der nichthydrostatischen Tsunamimodellierung. Mit Hilfe der in den Kapiteln 5 bis 8 erläuterten numerischen und algorithmischen Rechnungsverfahren soll nun die Rechenlaufzeit verringert werden. Um eine optimale Auslastung zu erzielen, werden die Stärken und Schwächen der verschiedenen Methoden betrachtet und deren Wechselwirkung untersucht.

9.1.1. Ablauf

Das Rechengitter wird zunächst für parallele Läufe partitioniert. Aus dieser Gebietszerlegung folgt die Verteilung des linearen Gleichungssystems, das iterativ gelöst wird. Um die Konvergenzrate zu beschleunigen, wird dieses globale Gleichungssystem in einen Satz von Teilsystemen zerlegt, welche lokal auf den einzelnen Prozessoren vorkonditioniert werden. Dieser Ablauf ist fix im Tsunamimodellierungsalgorithmus eingebettet. An verschiedenen Punkten kann nun eingegriffen werden und es gibt Wahlmöglichkeiten, welche Methode im Detail an dieser Stelle angewendet wird: Ausgehend von der Diskretisierung des Rechengebiets kann die Gitterknotennummerierung mit Hilfe von Permutationsalgorithmen umsortiert werden, wie in Abschnitt 5.2 beschrieben. Bei der Partitionierung können nasse und trockene Knoten unterschiedlich behandelt werden, um die Rechenlast möglichst gleichmäßig auf die Prozessoreinheiten zu verteilen. Dieses Vorgehen wird in Abschnitt 6.1.4 erläutert. Aus der Gebietszerlegung folgt die Verteilung des globalen Gleichungssystems. An dieser Stelle kann gewählt werden, ob stets das Gesamtsystem $\mathbf{Ax} = \mathbf{b}$ gelöst wird oder nur ein Teilsystem $\mathbf{A}_v \mathbf{x}_v = \mathbf{b}_v$, das sich auf die Werte des variablen Rechengebiets Ω_v beschränkt, siehe Kapitel 5.3. Um die Präkonditionierungsmethoden qualitativ miteinander vergleichen zu können, wird hier in allen Fällen der FGMRES-Algorithmus aus Kapitel 7 verwendet. Für das Zerlegen des globalen Systems in einen Satz Teilsysteme gibt es mehrere Ansätze, die in Kapitel 8.2 erläutert sind. Auch gibt es mehrere Varianten, welche Werte bei der unvollständigen LU-Zerlegung unterdrückt werden, um weiterhin mit dünnbesetzten Matrizen arbeiten zu können, siehe Abschnitt 8.1.2.

Da die gewählten Einstellungen sich gegenseitig beeinflussen können, wird zuerst mit dem einfachen Beispiel der stehenden Welle im Becken aus Abschnitt 4.1 gearbeitet, das nur eingeschränkte Wahlmöglichkeiten bietet. Anschließend werden die Untersuchungen auf die Simulation eines komplexen Tsunamiszenarios erweitert, das auf dem realen Ereignis beruht, welches sich 2010 vor der Küste Sumatras abspielte. Zur ef-

9. Effiziente Berechnungsmethoden

fizienten Berechnung stellen die beiden Beispiele unterschiedliche Anforderungen an die numerischen Verfahren.

9.1.2. Software

Im Rahmen dieser Arbeit wurde TsunAWI-NH geschaffen, das die Berechnung der vertikalen Geschwindigkeitskomponente und des nichthydrostatischen Druckterms, sowie die Korrektur der horizontalen Geschwindigkeitskomponenten ermöglicht. TsunAWI-NH ist eine Erweiterung von TsunAWI und kann einfach aktiviert oder deaktiviert werden. Um bestehende Löserbibliotheken einbinden zu können, deren parallele Berechnung auf MPI (Message Passing Interface) basiert, wurden für diese Arbeit die entsprechenden Implementierungen in TsunAWI/TsunAWI-NH durchgeführt, die eine Berechnung auf verteilten Systemen mit zugrundeliegender Gebietszerlegung des Rechengebiets ermöglichen. Somit wird bei der parallelen Berechnung kein Zugriff auf einen gemeinsamen Speicher für die Prozessoreinheiten vorausgesetzt und die Tsunamisimulation kann auch auf Clusterarchitekturen mit einer höheren Anzahl von Prozessoreinheiten durchgeführt werden. In der Vorprozessierung wird zur Gittergenerierung das externe Programm Triangle [60] verwendet, gefolgt von einer Relaxationsmethode nach [21], welche die Qualität des Gitters verbessert. Zur globalen Knotenvorsortierung werden die Matlab-Funktionen `randperm()`, `colamd()` und `symrcm()` [24] verwendet, um die Zufallssortierung wie die Permutationen COLAMD und RCM zu erzeugen. Die SFC-basierte Permutation wird mit einer am Rechenzentrum des Alfred-Wegener-Instituts bestehenden Routine erzeugt. Die Gebietszerlegung übernimmt das serielle Partitionierungsprogramm PaToH [13]. Um die daraus resultierende Verteilung zusammen mit der lokalen Vorsortierung für die Tsunamisimulation aufzubereiten, wird eine selbst implementierte hybride Routine ausgeführt, die sowohl auf MPI-Kommunikation, als auch auf OpenMP-Direktiven zurückgreift. Zum Lösen des in TsunAWI-NH auftretenden Gleichungssystems wird das Löserpaket pARMS 3.2 [38] verwendet. Da nicht alle verwendeten Optionen in dieser Version verfügbar sind, werden verschiedene selbst implementierte Routinen eingebettet. So zum Beispiel die Wiederverwendung der Indexmenge S und die Datenstrukturen der Dreiecksmatrizen \mathbf{L} und \mathbf{U} bei der levelbasierten unvollständigen LU-Faktorisierung, siehe Abschnitt 9.2.2. Auch der Ansatz Schur+RAS [37] als parallele Vorkonditionierung gehört zu dieser erweiterten Version.

9.1.3. Hardware

Die folgenden Rechenläufe wurden allesamt auf dem Clustersystem SGI-UV100 mit 1 bis 128 Intel Xeon E7-8837 Prozessorkernen à 2.66 GigaHertz des Alfred-Wegener-Instituts durchgeführt.

9.1.4. Kenngrößen

Die Tsunamisimulation besteht aus zwei Teilen: der Initialisierung und der Zeitintegration. Die Initialisierung umfasst das Importieren der Triangulierung samt Bathymetriedaten und Anfangsbedingungen. Nachbarschaftsbeziehungen und Ansatzfunktionen werden berechnet und Datenstrukturen angelegt. Die benötigte Zeit für diese einmaligen Aktionen wird mit t_{init} beschrieben. In der Zeitintegration wird Zeitschritt für Zeitschritt eine Momentaufnahme der Größen berechnet. Jeder der $n_{\Delta t} = t_{\text{end}}/\Delta t$ Zeitschritte ist gleich aufgebaut. Die Gesamtdauer der Zeitintegration t_{loop} wird im Folgenden als Vergleichswert verwendet. Das Schreiben der Ausgabedateien ist in diesem Kapitel bei der Zeitmessung ausgenommen, es wird nur der Rechenanteil betrachtet. In jedem Zeitschritt wird ein lineares Gleichungssystem gelöst, um den nichthydrostatischen Bodendruck zu bestimmen. Das Lösen des linearen Gleichungssystems wird in zwei Phasen aufgeteilt. Die Konfigurationsphase umfasst das Aufsetzen der Matrix und die unvollständige LU-Faktorisierung. Die Matrixeinträge wurden dabei schon im Vorfeld berechnet. Die Konfigurationsphase wird einmal pro Zeitschritt durchlaufen. In der Iterationsphase wird anschließend der iterative Lösungsalgorithmus FGMRES auf das Gleichungssystem angewandt, wobei in jedem Iterationsschritt die in der Konfigurationsphase berechnete Faktorisierung verwendet wird. Die Rechendauer für diese zwei Phasen wird als t_{konf} für die Konfigurations- und t_{iter} für die Iterationsphase bezeichnet. Die Anzahl der benötigten Iterationsschritte wird mit n_{it} angegeben. Dieser Wert ermöglicht es, die Qualität eines Prädiktionierers zu überprüfen. Je besser der Prädiktionierer, desto schneller konvergiert das Verfahren bezüglich der Iterationsanzahl. Die Werte t_{konf} , t_{iter} und n_{it} sind abhängig vom aktuellen Zeitschritt. In Abbildung 9.1 sind die Aufteilung von Rechenlauf und Gleichungssystem mit den zugehörigen Zeitspannen skizziert.

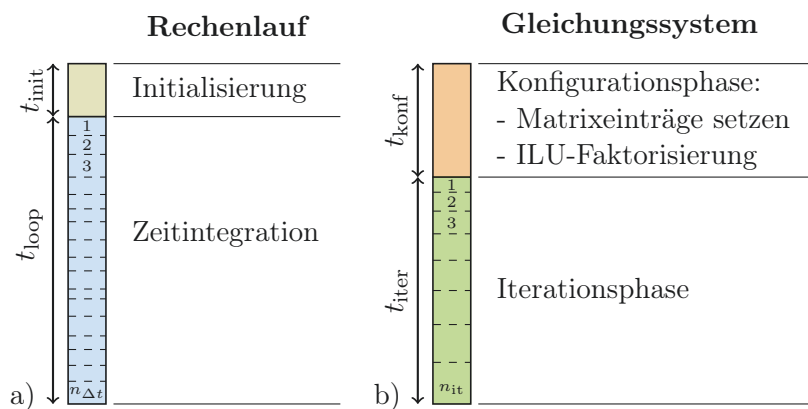


Abbildung 9.1.: Gemessene Zeitspannen a) des Gesamtrechenlaufs und b) der Berechnung des linearen Gleichungssystems eines Zeitschritts.

Die für einen einzelnen Iterationsschritt benötigte Zeit kann nicht direkt verglichen werden, da mit wachsender Dimension des Krylov-Unterraums die Anzahl der Rechenoperationen ansteigt. Deswegen wird eine über den Rechenlauf gemittelte Zeitspanne

9. Effiziente Berechnungsmethoden

$$\varnothing t_{\text{it}} := \frac{\sum_{k=1}^{n_{\Delta t}} t_{\text{iter}}(t^k)}{\sum_{k=1}^{n_{\Delta t}} n_{\text{it}}(t^k)} \quad (9.1)$$

mit $t^k = k\Delta t$ definiert. Werden auch n_{it} und t_{konf} über den gesamten Rechenlauf gemittelt, so werden diese Größen ebenfalls mit dem Symbol \varnothing gekennzeichnet. Die Gesamtlösezeit, die während eines Rechenlaufs mit dem Lösen der Gleichungssysteme verbraucht wird ist in

$$t_{\text{LGS}} := \sum_{k=1}^{n_{\Delta t}} \left(t_{\text{konf}}(t^k) + t_{\text{iter}}(t^k) \right) \quad (9.2)$$

zusammengefasst und entspricht einem großen Anteil von t_{loop} . Die Differenz

$$t_{\text{exp}} := t_{\text{loop}} - t_{\text{LGS}} \quad (9.3)$$

beschreibt demnach die Rechendauer für die expliziten Berechnungen innerhalb der Simulation. Beim Vergleich der verschiedenen Faktorisierungsansätze muss beachtet werden, dass je nach Variante unterschiedlich viele Werte unterdrückt werden. Als Vergleichswert wird im weiteren Verlauf der Füllfaktor

$$f_{LU} := \frac{|S|}{|S_A|} \quad (9.4)$$

herangezogen. Der Faktor f_{LU} beschreibt die Anzahl der Nichtnulleinträge der Dreiecksmatrizen, skaliert über die Mächtigkeit der Indexmenge S_A .

Bei parallelen Rechenläufen können ausgehend von der benötigten Rechenzeit die Beschleunigung

$$s_p := \frac{t_1}{t_p} \quad (9.5)$$

und die Effizienz

$$e_p := \frac{s_p}{p} \quad (9.6)$$

untersucht werden. Für die Beschleunigung s_p wird die Zeit t_1 des seriellen Laufs durch die Rechenzeit t_p geteilt, die beim entsprechenden parallelen Lauf mit p Prozessoren benötigt wird. Optimal wäre hier, wenn p Prozessoren nur den p -ten Teil von t_1 als Rechenzeit verwenden würden, also $s_p \equiv p$ gilt. Die Effizienz gibt an, wieviel Zeit ein Prozessor im Schnitt tatsächlich nutzt. Das Optimum ist der Verlauf $e_p \equiv 1$, so dass stets 100 Prozent der Zeit echt parallel gearbeitet wird. Dies ist in der Realität nicht möglich, da für den Datenaustausch Prozessoren aufeinander warten müssen. Darüber hinaus wird die Leistung durch dynamische Taktung und gemeinsame Nutzung der Hauptspeicherbandbreite der Multicore-Architektur der Xeon-Prozessoren gebremst. Beschränken sich die Untersuchungen auf das Lösen des Gleichungssystems, so werden im Folgenden Beschleunigung und Effizienz ausgehend von der Gesamtlösezeit t_{LGS} bestimmt. Wird auch die Zeit für den expliziten Rechenanteil beeinflusst, wie es zum Beispiel bei der Knotennummerierung der Fall ist, so kann gleichermaßen t_{exp} oder t_{loop} als Basiswert dienen.

9.2. Präkonditionierungsmethoden

In diesem Abschnitt werden die verschiedenen Präkonditionierungsmethoden anhand eines Testbeispiels und eines Tsunamiszenarios untersucht, wobei erst die lokalen ILU-Faktorisierungen und dann die globalen Vorkonditionierungsmethoden betrachtet werden. Zudem wird der Einfluss der globalen Gitterknotenanzahl auf die unvollständige LU-Faktorisierung geprüft.

Als Testlauf eignet sich das Beispiel der stehenden Welle im Becken, wie in Abschnitt 4.1 beschrieben. Dieses Beispiel hat den Vorteil, dass zu Beginn alle Knoten nass sind und während des Rechenlaufs kein einziger trocken fällt. Somit stimmen Ω_v und Ω ständig überein und es wird stets das Gesamtsystem berechnet. Da ausschließlich nasse Knoten existieren, ergibt sich eine homogene Rechenlast, so dass eine ungewichtete, einfache Partitionierung für die Gebietszerlegung ausreicht. Daten zu Länge und Breite des Beckens, sowie Wellenlänge und Amplitude sind in Tabelle 4.1 festgehalten. Als Beckentiefe wird $h_b = 5$ Meter gesetzt, so dass $h_b/\lambda = 0.25$ gilt. Um den parallelen Aspekt nicht außer Acht zu lassen, wird für einen erhöhten Rechenaufwand eine feinere Diskretisierung von Raum und Zeit verwendet, siehe Tabelle 9.1. Das verwendete Rechengitter besteht aus etwa zehntausend Gitterknoten. Es werden erst einige serielle Läufe untersucht. Im Anschluss wird das Verhalten bei parallelen Berechnungen betrachtet. Als Lösungsverfahren wird der FGMRES(15)-Algorithmus mit $\varepsilon = 10^{-12}$ verwendet. Bei der seriellen Berechnung entfällt die Wahl der globalen Vorkonditionierung, dafür wird statt einer Teilmatrix die Gesamtmatrix \mathbf{A} faktorisiert.

Tabelle 9.1.: Diskretisierungsparameter für das Testbeispiel der stehenden Welle.

	Parameter	Variable	Größe	Einheit
Raumdiskretisierung	Auflösung	Δx	~ 0.065	m
	Anzahl der Gitterknoten	N	10120	-
Zeitdiskretisierung	Zeitschrittweite	Δt	0.002	s
	Simulationsdauer	T_{end}	5.0	s
	Anzahl der Zeitschritte	$n_{\Delta t}$	2500	-

Die Einträge der Steifigkeitsmatrix \mathbf{A} aus (3.22) variieren mit der Zeitintegration. Sie müssen in jedem Zeitschritt neu berechnet und die Matrix entsprechend aufgesetzt werden. Die Indexmenge S_A , die das Füllmuster der Matrix \mathbf{A} definiert, wird von der Triangulierung und der Knotennummerierung des Rechengebiets bestimmt. Da sich diese innerhalb eines Rechenlaufs nicht verändern, ist über die Zeitintegration hinweg Strukturidentität gegeben. Die Position der Nichtnulleinträge bleibt erhalten, während sich die Werte der Einträge verändern. Das Anlegen der Matrixstruktur und das Freigeben des verwendeten Speichers benötigt Rechenzeit. Um für die Berechnung des Gesamtsystems $\mathbf{Ax} = \mathbf{b}$ diese Operationen nicht in jedem Zeitschritt zu wiederholen, werden sie einmalig zu Beginn beziehungsweise am Ende ausgeführt. Bei

9. Effiziente Berechnungsmethoden

der Zeitintegration werden die aktuellen Einträge der Matrix \mathbf{A} berechnet und an die entsprechenden Stellen geschrieben.

9.2.1. Faktorisierungsansätze

Ein serieller Testlauf wird nun mit unterschiedlichen Varianten unvollständiger LU-Zerlegungen durchgeführt. Die levelbasierten Varianten ILU(0), ILU(2) und ILU(3) stehen einer ILUT-Faktorisierung mit $\tau = 0.001$ und $n_f = 10$ gegenüber. Die Gitterknoten sind bei diesen Läufen mit dem SFC-Algorithmus vorsortiert. Der Einfluss anderer Knotenindizierungen wird in Abschnitt 9.2.4 untersucht.

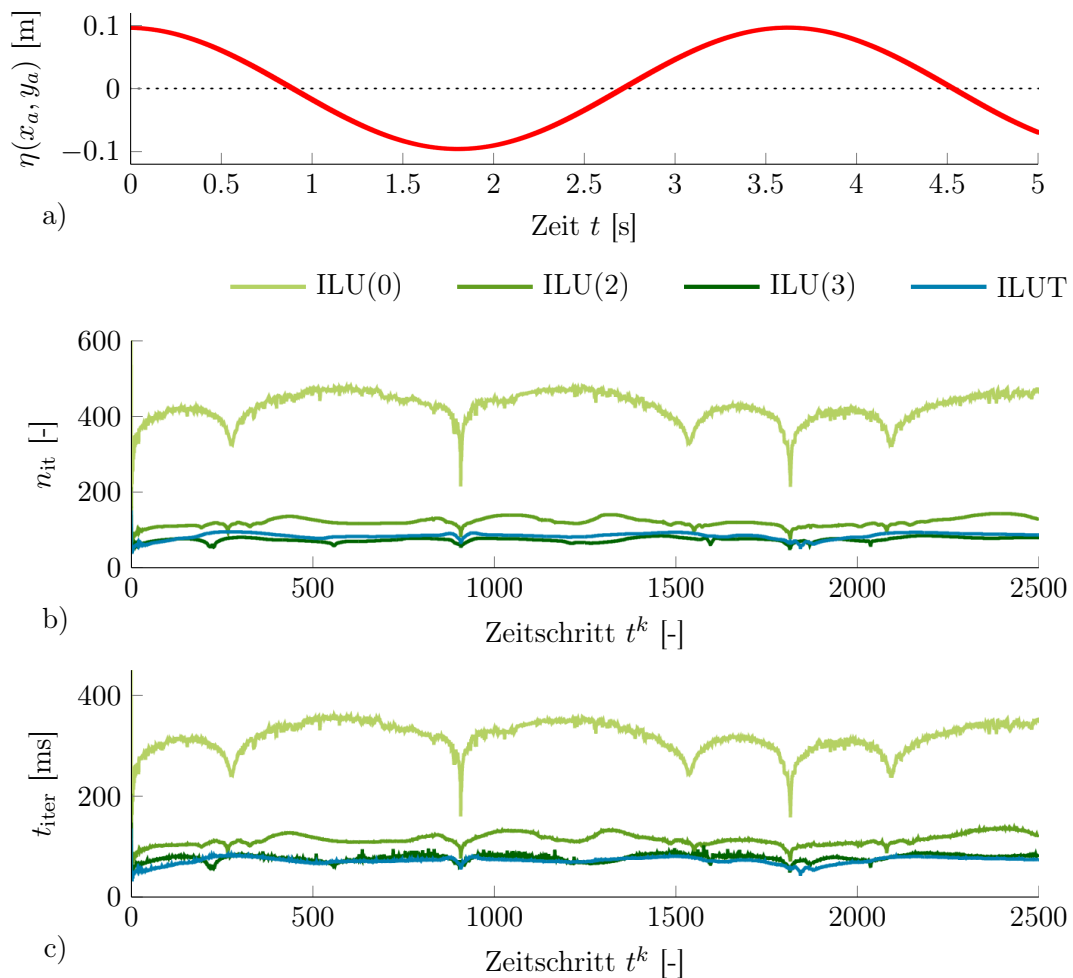


Abbildung 9.2.: a) Oberflächenauslenkung η am Punkt (x_a, y_a) ; b) Anzahl der Iterationsschritte n_{it} ; c) Anwendungszeit t_{iter} bei verschiedenen ILU-Varianten.

Abbildung 9.2 zeigt für die vier ILU-Varianten ILU(0), ILU(2), ILU(3) und ILUT die Zeit t_{iter} , die der Löser für die entsprechenden Anzahl von Iterationsschritten n_{it} je

Zeitschritt benötigt. Zur Orientierung ist in Abbildung 9.2 a) die Zeitserie der Oberflächenauslenkung η an einer beliebigen Position (x_a, y_a) aufgetragen. Die Anzahl der benötigten Iterationsschritte n_{it} verändert sich in allen Fällen mit der Wellenbewegung. Da als Startvektor \mathbf{x}_0 die Lösung des vorangegangenen Zeitschritts verwendet wird, konvergiert das Verfahren bei kleinen Veränderungen schneller. Bei den qualitativ hochwertigeren Methoden wie der ILU(3)- oder der ILUT-Faktorisierung schwankt n_{it} deutlich weniger, als bei der ILU(2)- oder der ILU(0)-Variante. Im allerersten Zeitschritt, wenn noch keine Näherung vorliegt, wird der Nullvektor als Startvektor verwendet. An dieser Stelle werden bei allen verwendeten ILU-Faktorisierungen die mit Abstand meisten Iterationsschritte benötigt. Der Vergleich der Bilder 9.2 b) und c) zeigt, dass die benötigte Zeit t_{iter} stark von der Iterationsanzahl n_{it} abhängt. Der Mittelwert $\varnothing t_{it}$ aus Gleichung (9.1) liefert also eine aufschlussreiche Information. Bei den levelbasierten Varianten sinken n_{it} und t_{iter} mit steigendem Level, während die ILUT-Faktorisierung in einer mit ILU(3) vergleichbaren Zeit eine größere Anzahl an Iterationsschritten berechnet. Dies hängt wiederum mit der Anzahl der Nichtnulleinträge innerhalb des Präkonditionierers, sowie mit besserer Datenlokalität zusammen. Je mehr Nichtnulleinträge in den Dreiecksmatrizen vorhanden sind, desto mehr Fließkommaoperationen müssen pro Iterationsschritt durchgeführt werden, was sich in der gemittelten Iterationszeit bemerkbar macht, siehe Abbildung 9.3.

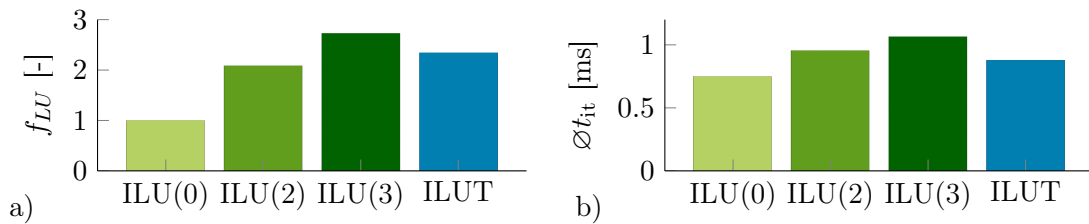


Abbildung 9.3.: a) Der Füllfaktor f_{LU} ; b) Die gemittelte Zeit pro Iterationsschritt $\varnothing t_{it}$ für die verschiedenen ILU-Faktorisierungen.

Auch wenn bei der ILUT-Faktorisierung der Füllfaktor f_{LU} während des Rechenlaufs variiert, wird hier nur ein Wert angegeben, da die Veränderung in der vierten Nachkommastelle stattfindet. Bei den levelbasierten Zerlegungen wird mit steigendem Level die Qualität des Präkonditionierers verbessert, da weniger Werte unterdrückt werden: n_{it} sinkt und $\varnothing t_{it}$ wächst. Die Wahl der Faktorisierungsvariante zeigt erheblichen Einfluss auf die Rechendauer. Die Algorithmen ILU(3) und ILUT benötigen zwar mehr Speicherplatz, führen jedoch zu einer schnelleren Konvergenz des FGMRES-Verfahrens, was sich durchaus positiv auf die Berechnungszeit auswirkt.

9.2.2. Datenstrukturen

Analog zur Matrix \mathbf{A} können die Datenstrukturen der Dreiecksmatrizen bei der unvollständigen LU-Faktorisierung wiederverwendet werden, vorausgesetzt die Indexmenge S_A kontrolliert das Füllmuster von \mathbf{L} und \mathbf{U} , wie es bei $ILU(k)$ geschieht. In diesem

9. Effiziente Berechnungsmethoden

Fall genügt es, die Indexmenge S einmal zu Beginn zu bestimmen. Anschließend werden beim Aufsetzen der Dreiecksmatrizen alle Werte unterdrückt, deren Indizes nicht in S enthalten sind. In pARMS 3.2 [38] wird die Wiederverwendung von S nicht unterstützt, deshalb wird eine Routine für levelbasierte Faktorisierungen implementiert, mit der die Strukturberechnung der Dreiecksmatrizen aus der Zeitintegration ausgeklammert wird. Die Vorgehensweise, wie die anfangs berechnete Füllstruktur bei der Faktorisierung verwendet wird, gleicht der Berechnung von $ILU(0)$ in Algorithmus 8.1, nur dass in der Abfrage S_A durch S ersetzt wird. Somit basiert die eingesparte Zeit bei der $ILU(0)$ -Zerlegung ausschließlich auf wiederverwendeten Datenstrukturen, während bei einer $ILU(k)$ -Zerlegung mit $k > 0$ zusätzlich der Faktorisierungsalgorithmus reduziert wird.

Für die Varianten $ILU(0)$, $ILU(2)$ und $ILU(3)$ wird nun der Testlauf der stehenden Welle wiederholt, wobei die Indexmenge S sowie die Matrixstrukturen wiederverwendet werden. Abbildung 9.4 a) zeigt die so gewonnene Zeiteinsparung in der Konfigurationsphase, wobei die Zeitspanne $\varnothing t_{\text{konf}}$ der $ILUT$ -Faktorisierung zum Vergleich aufgeführt wird. Werden Indexmenge und Datenstrukturen bei der Zeitintegration wiederverwendet, so findet die Berechnung der Indexmenge S und das Anlegen der Datenstrukturen in der Initialisierungsphase statt. Bei der $ILUT$ -Zerlegung ist die Berechnung der Indexmenge S von vornherein aufwendiger, vergleiche Algorithmus 8.1, und benötigt daher mehr Zeit. Die Datenstrukturen der Dreiecksmatrizen können nicht mehrfach genutzt werden, da die Indexmenge S nicht nur vom Füllmuster, sondern vor allem von den Werten der Matrix \mathbf{A} abhängt. Dass die Läufe mit den wiederverwendeten Datenstrukturen nicht alle dieselbe Zeit für die Konfiguration aufwenden, liegt daran, dass je nach ILU -Variante unterschiedlich viele Werte berechnet werden, wie im vorherigen Abschnitt gezeigt.

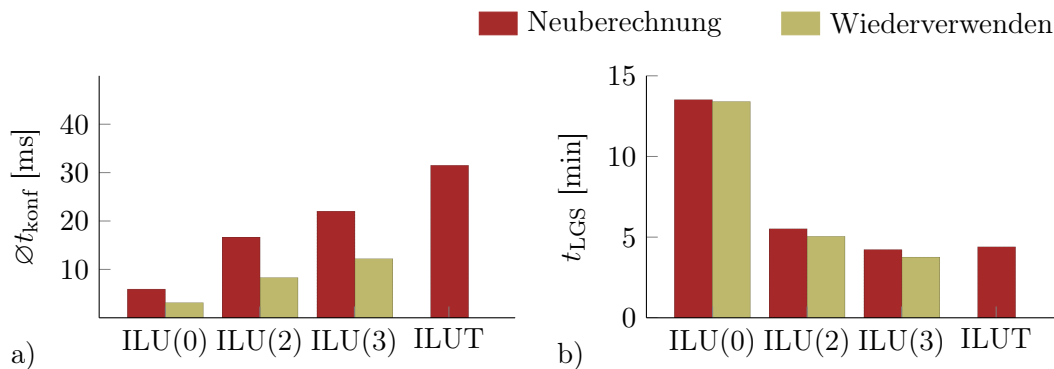


Abbildung 9.4.: a) Mittlere Konfigurationszeit $\varnothing t_{\text{konf}}$ und b) Gesamtlösezeit t_{LGS} bei Neuberechneten und wiederverwendeten Datenstrukturen.

Durch die mehrfache Nutzung von Datenstrukturen wird die Konfigurationsphase erheblich verkürzt. Welche Auswirkung dies auf die Gesamtlösezeit t_{LGS} hat, hängt davon ab, welchen Anteil t_{konf} im Vergleich zu t_{iter} in Gleichung (9.2) einnimmt. In diesem Beispiel überwiegt jedoch der Zeitaufwand in der Iterationsphase wie in Abbildung 9.4 b) im Vergleich der Gesamtlösezeit t_{LGS} deutlich zu sehen ist. Da die Anwen-

dungsphase pro Zeitschritt viel mehr Rechenzeit benötigt als die Konfigurationsphase, ist der Unterschied zwischen den Läufen mit Neuberechneten beziehungsweise wiederverwendeten Datenstrukturen der Faktorisierung nicht so drastisch wie im Vergleich von t_{konf} . Allerdings muss beachtet werden, dass je schneller der Algorithmus konvergiert, desto mehr Einfluss gewinnt die Konfigurationsphase auf die Gesamtlösezeit. Auch wenn ILUT und ILU(3) bei Neuberechneten Datenstrukturen vergleichbare Zeiten liefern, so ist die Zeiteinsparung bei einer Wiederverwendung der Strukturen hoch genug, dass die Levelvariante klar im Vorteil liegt. Falls möglich, werden für weitere Untersuchungen die Datenstrukturen bei der unvollständigen LU-Berechnung zu Beginn angelegt und mit Hilfe der in der Initialisierungsphase berechneten Indexmenge S während des Rechenlaufs wiederverwendet.

9.2.3. Vergleich der Methoden zur parallelen Vorkonditionierung

Nun wird das Testproblem der stehenden Welle parallel auf zwei, vier und acht Prozessorkernen gerechnet, wobei verschiedene Ansätze zur parallelen Vorkonditionierung betrachtet werden, unter anderem das Block-Jacobi-Verfahren (BJ) und die Restricted Additive Schwarz Methode (RAS). Darüber hinaus wird der Ansatz untersucht, bei dem ein approximiertes Schursystem gelöst wird und mit dessen Resultat die übrigen Werte explizit bestimmt werden. Für dessen Vorkonditionierung wird einmal BJ auf die globale Schurmatrix angewandt und das Gleichungssystem mit einem inneren GMRES-Verfahren gelöst (Schur+GMRES), wobei maximal fünf Iterationsschritte erlaubt sind und das Toleranzkriterium $\|\mathbf{r}_m\| < \varepsilon \|\mathbf{r}_0\|$ mit $\varepsilon = 0.001$ gilt. Als zweite Variante wird RAS auf das Schursystem angewandt (Schur+RAS). Dieser letzte Ansatz wird von pARMS 3.2 nicht unterstützt. Motiviert von [37] wurden die entsprechenden Routinen von der Autorin implementiert und eingebettet. Diese Implementierung der Schur+RAS-Vorkonditionierung setzt voraus, dass die Belegungsstruktur des approximierten Schurkomplements \mathbf{S} wiederverwendet wird, da andernfalls die Konfigurationszeit t_{konf} außergewöhnlich hoch ist. Da \mathbf{S} mittels vorzeitig abgebrochener, unvollständiger LU-Faktorisierung generiert wird, ist die Kombination Schur+RAS/ILUT nicht möglich. Beim Lösen eines verteilten Gleichungssystems wird die Arbeit parallel durchgeführt, was die Rechenzeit senkt. Allerdings verschlechtert sich die lokale Präkonditionierung mittels unvollständiger LU-Zerlegung, da nur noch eine Teilmatrix der Matrix \mathbf{A} in die Faktorisierung miteinfließt. Da die ILU(0)-Variante im seriellen Lauf sehr schlecht abgeschnitten hat, wird diese Variante hier nicht mehr aufgeführt.

In Tabelle 9.2 ist festgehalten, welche der globalen Präkonditionierungsmethoden bei Konfiguration und Anwendung Daten austauschen. Die Art des Datenaustauschs ist dabei in die Verwendung von 1:1-Kommunikationsaufrufen und kollektiven Kommunikationsroutinen aufgeschlüsselt. Bei kollektiven Kommunikationsroutinen sind alle Prozessoreinheiten beteiligt, während bei einer 1:1-Kommunikation eine PE Daten direkt an eine zweite versendet, welche diese dann empfängt. Bei der RAS-Methode werden zum Beispiel sowohl in der Konfigurations- als auch in der Anwendungsphase zwischen einzelnen Prozessoren Daten ausgetauscht, während BJ in keiner Phase Daten synchronisiert. Datenaustausch kann eine parallele Rechnung stören, da Prozessoren gegebenenfalls aufeinander warten müssen, bevor weitergerechnet werden kann.

9. Effiziente Berechnungsmethoden

Bei einer 1:1-Kommunikation wartet dabei höchstens ein Prozessor, bei einer kollektiven Kommunikationsroutine schlechtestenfalls $p - 1$ Prozessoren auf einen langsamen.

Tabelle 9.2.: Datenaustausch während Konfiguration und Anwendung bei unterschiedlich Präkonditionierungsmethoden.

Phase	Datenaustausch	BJ	rAS	Schur + GMRES	SchurRAS
Konfiguration	1:1	-	✓	-	✓
	kollektiv	-	-	-	-
Anwendung	1:1	-	✓	-	✓
	kollektiv	-	-	✓	-

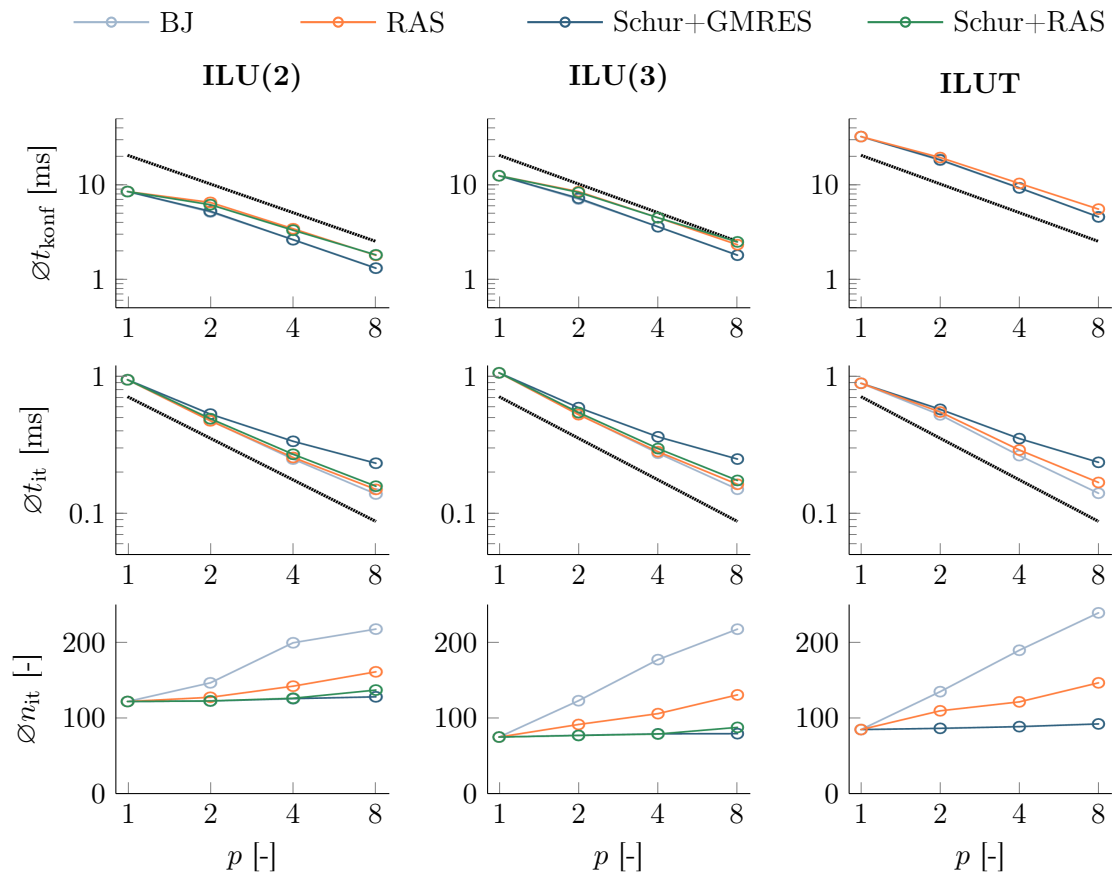


Abbildung 9.5.: Die gemittelten Kenngrößen $\varnothing t_{\text{konf}}$, $\varnothing t_{\text{it}}$ und $\varnothing n_{\text{it}}$ für verschiedene Vorkonditionierungskombinationen. Die schwarze Linie zeigt die ideale Steigung.

Abbildung 9.5 zeigt die Veränderung der gemittelten Kenngrößen $\varnothing n_{it}$, $\varnothing t_{it}$ und $\varnothing t_{konf}$ bei wachsender Prozessorzahl p für verschiedene Kombinationen von paralleler Vorkonditionierung und ILU-Varianten. Es ist leicht zu erkennen, dass die Kurven bezüglich der parallelen Vorkonditionierung in allen drei Spalten ähnlich verlaufen. Das Verhalten der parallelen Präkonditionierungsverfahren ist bei ansteigender Prozessorzahl demnach unabhängig von der ILU-Variante. Quantitativ gibt es natürlich Unterschiede zwischen den Kombinationen. In den gemittelten Zeiten $\varnothing t_{konf}$ und $\varnothing t_{it}$ ist deutlich der Kommunikationsaufwand in Konfigurations- und Berechnungsphase der einzelnen parallelen Verfahren zu erkennen. Die Qualität der Methode lässt sich in der gemittelten Anzahl der Iterationsschritte $\varnothing n_{it}$ ablesen. Es fällt auf, dass das Lösen des approximierten Schursystems sich als Vorkonditionierung besser eignet, als die direkte Vorkonditionierung des Gesamtsystems. Dies wird vor allem im Vergleich von RAS und Schur+RAS deutlich. Dieselben Operationen führen zu schnellerer Konvergenz, wenn der Umweg über das Schursystem gegangen wird. Konfiguration und Anwendung entsprechen dem gleichen Zeitaufwand, doch n_{it} steigt bei der Vorkonditionierung des Gesamtsystems viel stärker an. Auch das GMRES-Verfahren als Alternative zu RAS zum Lösen des Schursystems hat vergleichbar wenige Iterationsschritte nötig. Bei acht Prozessoren sind kaum mehr Iterationen nötig als beim seriellen Rechenlauf. Das Abknicken der durchschnittlichen Rechenzeit pro Iterationsschritt bei dieser Methode ist auf die globalen Skalarprodukte im GMRES-Algorithmus zur Generierung der Orthonormalbasis des Krylov-Unterraums innerhalb des Präkonditionierers zurückzuführen. Im Gegensatz zur Matrix \mathbf{A} ist die Schurmatrix ungleichmäßig verteilt, wie aus der in Abbildung 9.6 skizzierten Verteilung von Gitterknoten und lokaler Grenzknoten hervorgeht. Es herrscht ein Lastungleichgewicht. Bei einem kollektiven Kommunikationsaufruf, wie es beim Skalarprodukt der Fall ist, müssen Prozessoren mit wenig Rechenlast auf andere Prozessoren warten. Bei der Schur+RAS-Methode werden zwar ebenfalls Daten ausgetauscht, jedoch nicht so häufig und 1:1 statt kollektiv. Dafür braucht die (Schur+)RAS-Methode allgemein mehr Zeit bei der Konfiguration, da auch hier Daten ausgetauscht werden müssen.

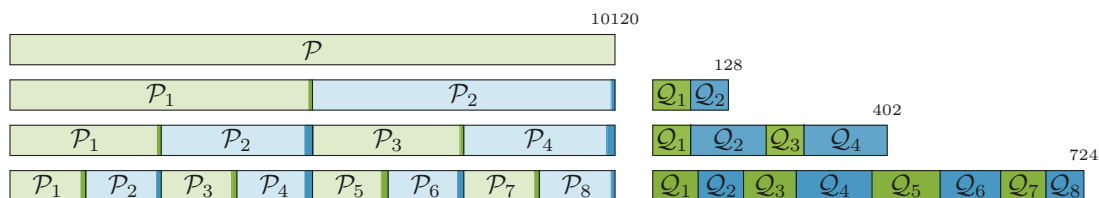


Abbildung 9.6.: Verteilung der Gitterknoten (links) und der lokalen Grenzknoten (rechts) bei verschiedenen Zerlegungen. Rechts oben steht jeweils die entsprechende Knotenanzahl.

Das Block-Jacobi-Verfahren legt ein ähnliches Verhalten an den Tag wie zuvor ILU(0) im Vergleich der verschiedenen Faktorisierungsvarianten: Konfiguration und Berechnung eines Iterationsschritts sind sehr effizient, während die benötigte Iterationsanzahl sehr hoch ist (Die entsprechende Kurve bezüglich $\varnothing t_{konf}$ ist identisch mit jener von Schur+GMRES). Zur Erinnerung: Bei BJ werden diejenigen Einträge der Matrix \mathbf{A}

9. Effiziente Berechnungsmethoden

bei der Prädiktionierung ignoriert, die nicht in den Diagonalblöcken angesiedelt sind. So werden weder beim Konfigurieren noch bei der Anwendung Daten zwischen den Prozessoreinheiten ausgetauscht und eine parallel unabhängige Vorkonditionierung ist möglich. Allerdings ist die Annäherung an \mathbf{A}^{-1} dürftig, so dass auch bei einer qualitativ hochwertigen ILU-Variante sehr viele Iterationsschritte nötig sind, bis das Konvergenzkriterium erreicht ist.

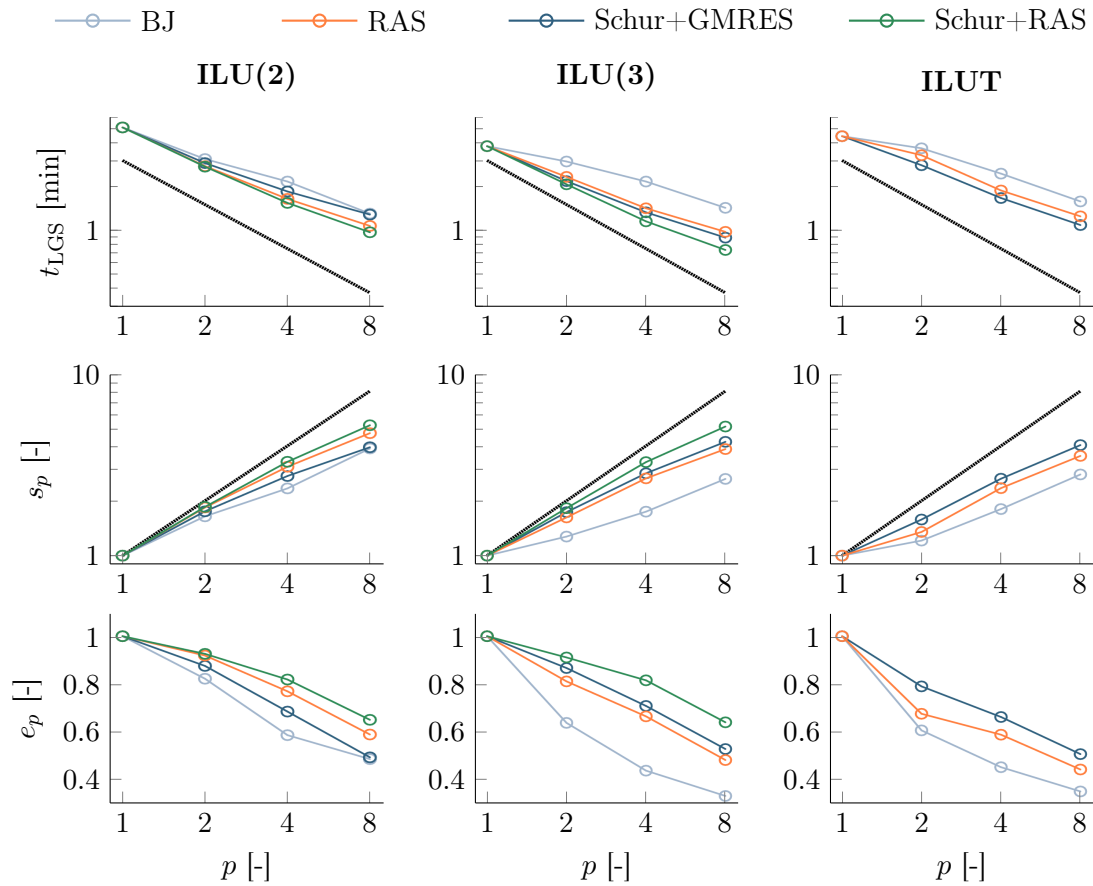


Abbildung 9.7.: Gesamtlösezeit t_{LGS} , Beschleunigung s_p und Effizienz e_p für verschiedene Vorkonditionierungskombinationen. Die schwarze Linie zeigt die ideale Steigung.

Unter Berücksichtigung der Gesamtlösezeit t_{LGS} , einschließlich der davon abgeleiteten Beschleunigung s_p und Effizienz e_p , dargestellt in Abbildung 9.7, kann das Block-Jacobi-Verfahren zur globalen Vorkonditionierung des Gesamtsystems für weitere Untersuchungen ausgeschlossen werden. Die drastische Verschlechterung des globalen Prädiktionierers und die resultierende hohe Iterationsanzahl n_{it} bei der parallelen Berechnung schließen eine effiziente Lösung des Gleichungssystems aus. Die Schur+RAS-Methode führt in Kombination mit den levelbasierten LU-Zerlegungen zu den besten Ergebnissen, während für RAS und Schur+GMRES keine allgemeine Qualitätsaussage getroffen werden kann. Im direkten Vergleich der beiden Verfahren

führt die erhöhte Iterationszeit, die Schur+GMRES aufgrund der kollektiven Kommunikationsroutinen aufbringt, bei einer ILU(2)-Zerlegung zu einem Nachteil. Dies kann jedoch bei einer besseren Faktorisierung mit der geringen Anzahl von Iterationsschritten und einer kürzeren Konfigurationszeit kompensiert werden.

9.2.4. Einfluss der Gitternummerierung

Mit einer Umsortierung der Gitternummerierung soll nun die LU-Faktorisierung verbessert werden, siehe Abschnitt 5.2. Des Weiteren kann sich die resultierende Datenlokalität als cache-effizient erweisen, da weniger Daten aus dem Hauptspeicher nachgeladen werden müssen. Die Zufallspermutation ist als Repräsentant einer unsortierten Knotenanordnung gedacht. Obwohl sich ILU(0) in Abschnitt 9.2.1 im Vergleich zu anderen ILU-Varianten als ineffizient erwiesen hat, wird diese Art der Faktorisierung hier noch einmal aufgegriffen, da in diesem Fall laut Definition stets das Verhältnis $f_{LU} = 1$ gilt. Mit gleichen Voraussetzungen kann die Auswirkung der Umsortierung auf die Qualität des Präkonditionierers direkt gezeigt werden. Als Alternative zur SFC-Sortierung für die Gitterknoten werden hierbei Zufalls-, RCM- und COLAMD-Permutationen verwendet. Ausgehend von der Knotennummerierung werden die Elemente Ω^e in numerisch aufsteigender Reihenfolge des kleinsten Knotenindizes je Element sortiert. Die entsprechende Anordnung der Gitterknoten im Rechengitter ist in Abbildung 9.8 farblich dargestellt.

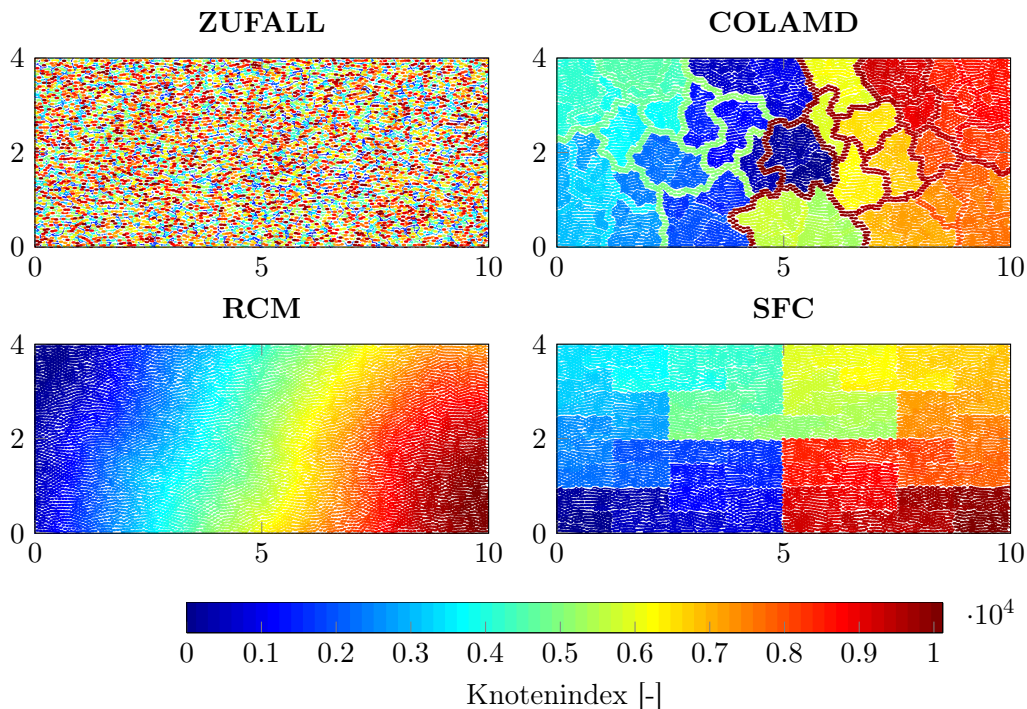


Abbildung 9.8.: Globale Knotennummerierung bei verschiedenen Sortierungen.

9. Effiziente Berechnungsmethoden

Zuerst werden die seriellen Läufe für die verschiedenen ILU-Varianten wiederholt, wobei die Gitterknoten und -elemente in einer anderen Reihenfolge angeordnet sind. Im Anschluss wird das Verhalten bei wachsender Prozessorzahl exemplarisch an einer Vorkonditionierungskombination gezeigt. Tabelle 9.3 zeigt die Füllfaktoren f_{LU} der unterschiedlichen ILU-Faktorisierungen bei den vier Knotenindizierungen bei serieller Berechnung. Bei den $ILU(k)$ -Zerlegungen erzeugt die Zufallspermutation den höchsten Füllfaktor, während die RCM-Sortierung weniger Werte verwendet. Bei einer ILUT-Faktorisierung ist es genau umgekehrt. COLAMD und SFC liegen unabhängig von der unvollständigen Zerlegung mit ähnlichen Faktoren dazwischen.

Tabelle 9.3.: Der Füllfaktor f_{LU} je nach Kombination von Knotensortierung und ILU-Variante.

	ILU(0)	ILU(2)	ILU(3)	ILUT
ZUFALL	1.00	2.45	3.29	2.21
COLAMD	1.00	2.09	2.76	2.37
RCM	1.00	1.81	2.35	2.80
SFC	1.00	2.08	2.72	2.34

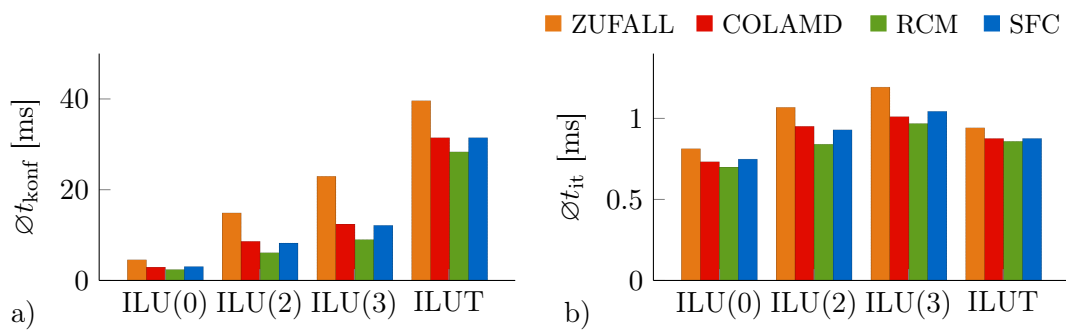


Abbildung 9.9.: Mittlere Konfigurations- und Iterationszeit bei unterschiedlichen Knotensortierungen.

Der Vergleich der gemittelten Zeitspannen $\varnothing t_{\text{konf}}$ und $\varnothing t_{\text{it}}$ in Abbildung 9.9 zeigt, dass nicht allein die Anzahl der Matrixeinträge, sondern in Bezug auf Datenlokalität auch die Position der Einträge eine Rolle spielt. Ein höherer Füllfaktor bewirkt nicht zwangsläufig eine größere Zeitspanne. Die Zufallspermutation verursacht auch bei kleinem f_{LU} erhöhten Zeitaufwand bei Konfiguration und Berechnung, während bei einer RCM-Sortierung die ILUT-Zerlegung trotz hohem f_{LU} schnell konfiguriert ist. Die durchschnittliche Anzahl der benötigten Iterationsschritte pro Zeitschritt ist für die verschiedenen ILU-Varianten in Abbildung 9.10 a) aufgetragen. Die RCM-Sortierung bringt hier eindeutig die beste Vorkonditionierung hervor, während die Berechnung basierend auf der Zufallspermutation etwa doppelt so viele Iterationen zur Lösung der Gleichungssysteme aufwendet. Der hohe Füllfaktor f_{LU} verursacht mehr Fließkommaoperationen in den Iterationsschritten und Mehraufwand bei der Konfiguration, führt jedoch nicht zu einer besseren Qualität des Präkonditionierers. Bei

der ILU(0)-Variante führt die COLAMD-Sortierung verglichen mit SFC schneller zu einem Ergebnis, bei den übrigen Läufen erzeugt die SFC-Sortierung den besseren Prädiktionierer. Da die Gitternummerierung nicht nur die ILU-Zerlegung, sondern auch durch Datenlokalität die gesamte Berechnung beeinflussen kann, ist in Abbildung 9.10 b) t_{loop} aufgetragen, wobei der dunkel eingefärbte Anteil t_{exp} dem expliziten Rechenanteil entspricht. Dieser Anteil ist in diesem Beispiel sehr gering und t_{exp}/t_{loop} liegt bei einer ILU(0)-Faktorisierung stets unter 6.5 Prozent, bei den übrigen Kombinationen unter 17 Prozent. Dennoch wird sichtbar, dass die Zufallspermutation auch für diesen Anteil mehr Zeit benötigt. Der helle Anteil t_{LGS} spiegelt den hohen Einfluss der Iterationsanzahl im linken Bild wider.

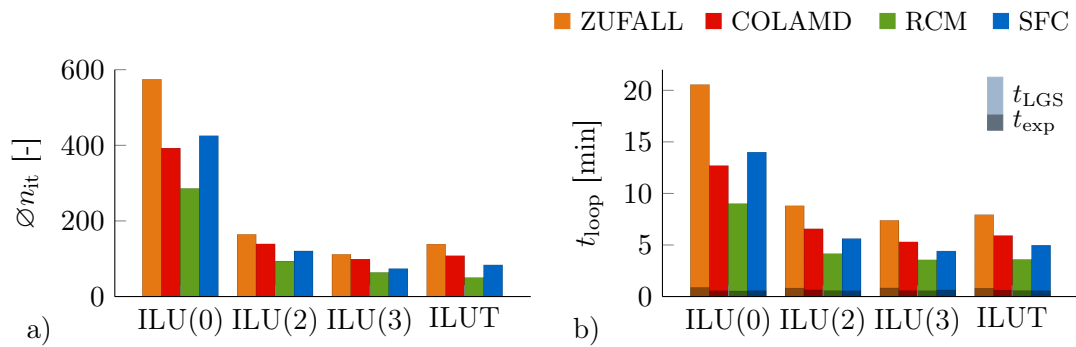


Abbildung 9.10.: a) Die gemittelte Iterationsanzahl \bar{n}_{it} ; b) Die Gesamtlaufzeit t_{loop} mit hell hervorgehobenem Anteil der Gesamtlöserzeit t_{LGS} .

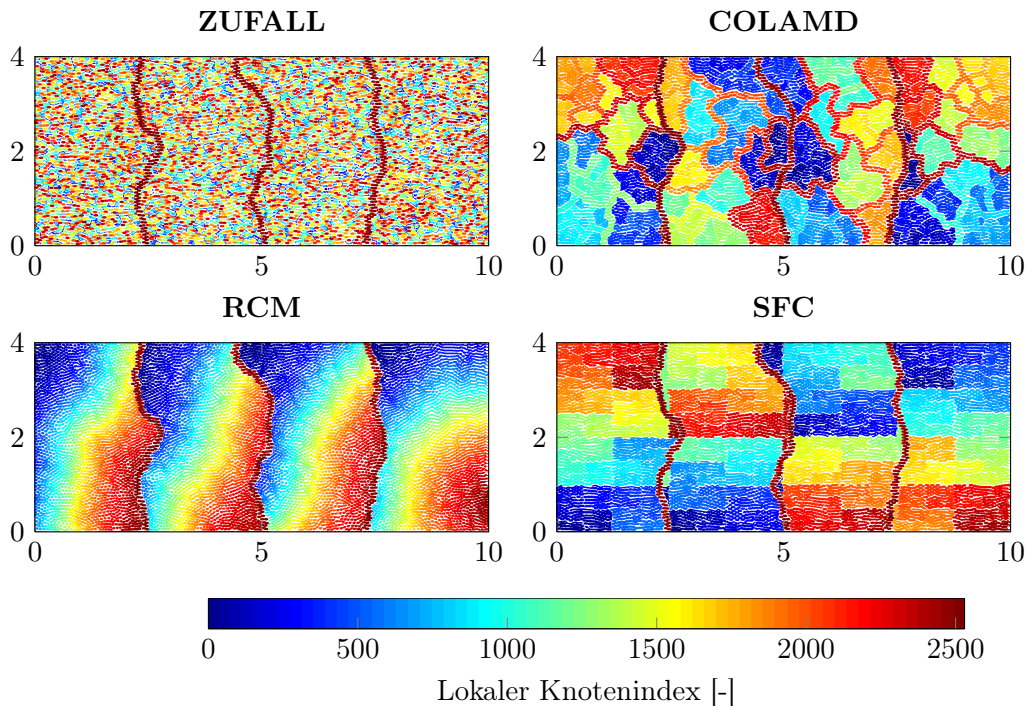


Abbildung 9.11.: Lokale Knotennummerierung bei verschiedenen Permutationen.

9. Effiziente Berechnungsmethoden

Die Permutationsalgorithmen legen die globale Nummerierung der Knoten fest. Bei der Partitionierung erhält jede Prozessoreinheit jedoch nur eine Teilmenge dieser Knoten. Diese Teilmenge wird noch einmal lokal umsortiert, indem zwischen inneren Knoten und Grenzknoten unterschieden wird, siehe Abschnitt 6.2.1. Die größere Gruppe der inneren Knoten wird jedoch in aufsteigender Reihenfolge der globalen Knotennummerierung angeordnet, so dass wie in Abbildung 9.11 die globale Sortierung klar erkennbar bleibt.

Abbildung 9.12 zeigt deutlich den Einfluss, den die Gitternummerierung auf die Präkonditionierung ausübt. Mit einer RCM-Sortierung konvergiert das Krylov-Unterraumverfahren am schnellsten. Dies wirkt sich auch auf die Gesamtlösezeit t_{LGS} aus. Die Rechenzeit t_{exp} zeigt, inwiefern sich die Anordnung auf die expliziten Berechnungen auswirkt. Abgesehen von der Zufallspermutation sind die Unterschiede hier nicht sehr hoch, dennoch führt auch hier die RCM-Sortierung.

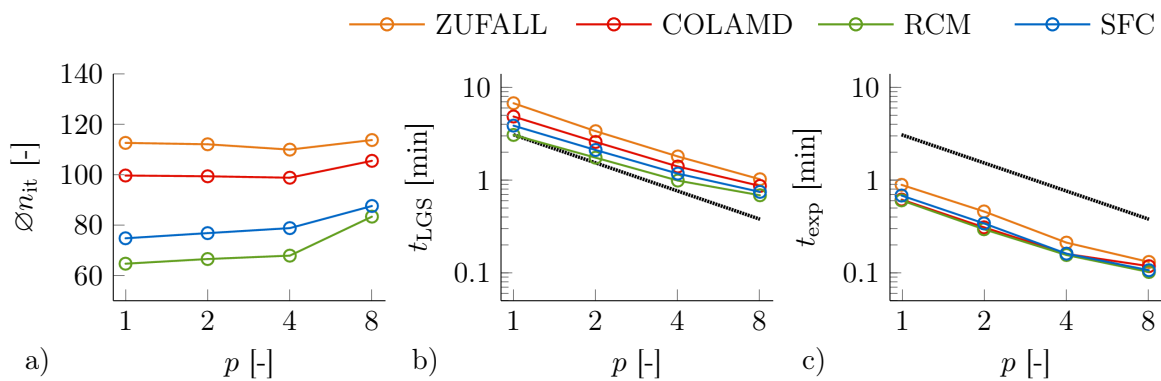


Abbildung 9.12.: Die gemittelte Iterationsanzahl $\varnothing n_{it}$, sowie die Rechenzeiten t_{LGS} und t_{exp} bei unterschiedlichen Gitternummerierungen mit einer Schur+RAS/ILU(3)-Vorkonditionierung.

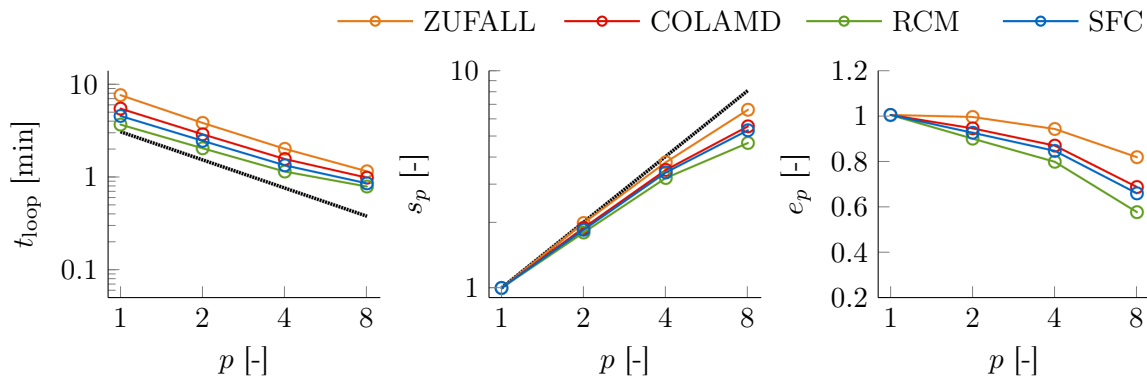


Abbildung 9.13.: Die Rechendauer t_{loop} , sowie die Beschleunigung s_p und die Effizienz e_p bei unterschiedlichen Gitternummerierungen mit einer Schur+RAS/ILU(3)-Vorkonditionierung.

In Abbildung 9.13 sind exemplarisch für die Kombination Schur+RAS/ILU(3) die Berechnungszeit t_{loop} und davon abgeleitet, die Beschleunigung s_p und die Effizienz e_p eingezeichnet. Bei einer zufällig geordneten Knotenindizierung ermöglicht die zerlegte Matrix mit lokal umsortierten Knoten eine bessere Faktorisierung als die Zerlegung der Gesamtmatrix. Die Beschleunigung s_p liegt teilweise sogar über dem Optimum. In diesem Fall wird von einer superlinearen Beschleunigung gesprochen. Die Vorteile der anderen Knotensortierungen werden mit ansteigender Prozessoranzahl unterschiedlich stark gestört. Im Vergleich der Effizienz wird deutlich, dass die Prozessoren bei einer Zufalls- oder COLAMD- zwar besser ausgelastet sind, jedoch bezogen auf die Rechenzeit t_{loop} eine schlechte Wahl darstellen. Der Einfluss der verschiedenen Gitternummerierungen ist durchaus sichtbar, doch sehr gering, verglichen mit der Wahl der Vorkonditionierung. Dennoch lohnt es sich auf jeden Fall zu sortieren. Eine zufällige Nummerierung erzeugt bei der Faktorisierung trotz hoher Anzahl von Nichtnulleinträgen keine gute Vorkonditionierung. Zudem ist diese Anordnung nicht cache-effizient. Mit der RCM-Vorsortierung konvergiert das Krylov-Unterraum weitaus schneller als die getesteten Alternativen, allerdings verringert sich der Unterschied zur SFC-Permutation bei $p = 8$. Das im nächsten Abschnitt behandelte Tsunamiszenario wird mit bis zu 128 Partitionen gerechnet, so dass sich zeigt, wie sich die Störung der globalen Vorsortierung bei größeren Prozessoranzahlen auf die Gesamtrechenzeit auswirkt.

9.2.5. Vorkonditionierung beim Tsunamiszenario

Nun wird ein Teil der vorigen Untersuchungen bei der Berechnung eines Szenarios wiederholt, das auf einem realen Tsunamireignis beruht. Die Voraussetzungen für eine effiziente Berechnung sind hier anders als beim Beispiel der stehenden Welle im Becken. Der dynamische Bodendruck ändert sich nur in einem Teil des Rechengebiets. Viele der Gitterknoten liegen an Land, so dass der Rechenaufwand nicht an allen Knoten gleich hoch ist, was zu einem Lastungleichgewicht führt.

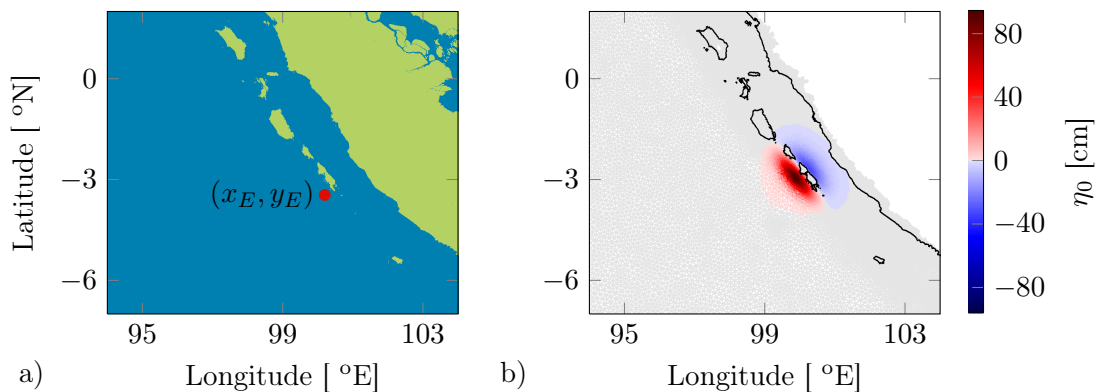


Abbildung 9.14.: a) Koordinaten des Epizentrums; b) Anfangsauslenkung η_0 für ein Beben mit der Magnitude $M_w = 7.8$.

9. Effiziente Berechnungsmethoden

Das hier gerechnete Szenario basiert auf dem Tsunami, der am 25. Oktober 2010 durch ein Seebeben der Stärke $M_w = 7.8$ angeregt wurde. Das Epizentrum des Bebens wurde mit den Koordinaten (100.2 °E, -3.46 °N) nahe der Mentawai-Inselkette vor der Westküste Sumatras lokalisiert [29]. Dieses Beben verursachte einen lokalen Tsunami, der mehr als vierhundert Menschenleben forderte.

Ausgehend von diesen Daten wird mit RuptGen 2.1 [6] eine Initialauslenkung η_0 generiert, die direkt auf die Gitterkoordinaten interpoliert wird. Als Anfangsgeschwindigkeit wird $\mathbf{u}_0 = 0$ gesetzt. In Abbildung 9.14 a) ist das Epizentrum eingezeichnet, 9.14 b) zeigt die mit RuptGen erzeugte Anfangsauslenkung auf dem Rechengitter. Bei einer Zeitschrittweite von einer Sekunde werden die ersten zwei Stunden nach dem Beben simuliert. Details zur Raum- und Zeitdiskretisierung des Mentawai-Szenarios sind in Tabelle 9.4 zusammengefasst.

Tabelle 9.4.: Diskretisierungsdaten zum Mentawai-Szenario.

	Parameter	Variable	Größe	Einheit
Raumdiskretisierung	Gittertyp		regional	
	Auflösung	Δx	[0.054, 22.5]	km
	Knotenanzahl	N	629061	-
Zeitdiskretisierung	Zeitschrittweite	Δt	1.0	s
	Simulationsdauer	T_{end}	2.0	h
	Zeitschrittanzahl	$n_{\Delta t}$	7200	-

Wie beim Beispiel der stehenden Welle wird das Gesamtsystem gelöst. Bei der hier verwendeten Partitionierung werden trockene Knoten nicht gesondert behandelt. Da die ILU(0)-Faktorisierung sich als ineffizient erwiesen hat, wird diese Zerlegung hier nicht verwendet. Es werden die Permutationen COLAMD, RCM und SFC für die Gitternummerierung benutzt.

Tabelle 9.5.: Der Füllfaktor f_{LU} abhängig von der ILU-Variante und der verwendeten Permutation.

	ILU(2)	ILU(3)	ILUT
COLAMD	2.11	2.83	0.18
RCM	1.84	2.42	0.18
SFC	2.11	2.78	0.18

Die Existenz trockener Knoten wird im Füllfaktor f_{LU} der ILUT-Zerlegung wiedergegeben, siehe Tabelle 9.5. Die Behandlung der trockenen Knoten im Gesamtsystem führt zu vielen Nulleinträgen in der Matrix \mathbf{A} , die wegen der mehrfach genutzten Ma-

trixstruktur gespeichert werden. Da bei der ILUT-Faktorisierung alle Einträge kleiner τ unterdrückt werden, wird die Belegungsstruktur ausgedünnt. Bei den ILU(k)-Varianten werden die gespeicherten Nullen der Matrix \mathbf{A} auch bei der Zerlegung mitberücksichtigt, die Füllfaktoren sind vergleichbar mit denen aus Tabelle 9.3.

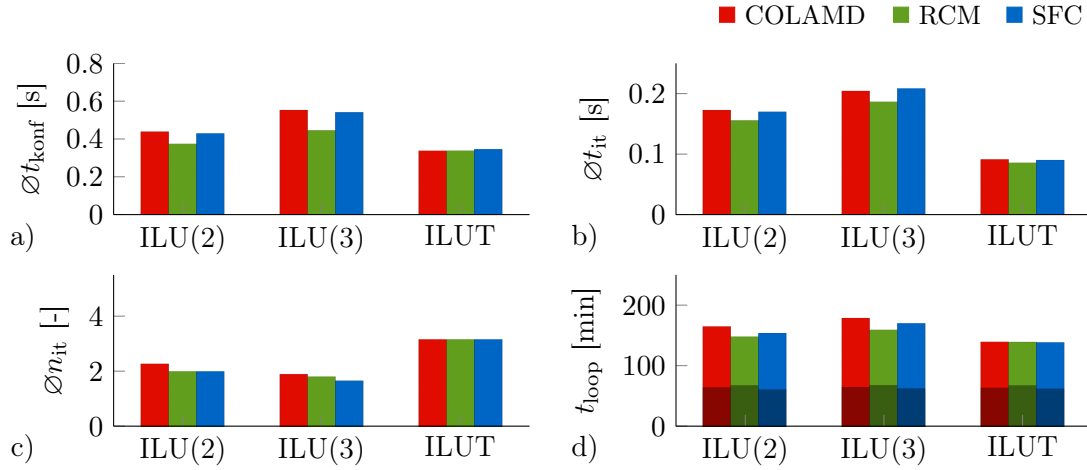


Abbildung 9.15.: Die gemittelten Werte $\varnothing t_{\text{konf}}$, $\varnothing t_{\text{it}}$, $\varnothing n_{\text{it}}$ sowie die Gesamtlaufzeit t_{loop} mit hell hervorgehobenem Löseranteil t_{LGS} .

Auch wenn die ILUT-Zerlegung Zeitverluste durch die Neuberechnung der LU-Strukturen zu verzeichnen hat, so hat sie hier einen Zeitbonus im Vergleich zu den levelbasierten Faktorisierungen, die über das Zehnfache an Werten berechnen und einsortieren müssen. Die Unterschiede bezüglich der Konfigurationszeit im seriellen Lauf, dargestellt in Abbildung 9.15 a) sind viel geringer als bei der stehenden Welle, siehe 9.9 a). Die Berechnungszeit $\varnothing t_{\text{it}}$ wird wie im vorigen Abschnitt mehr von der Faktorisierungsvariante, als von der Knotennummerierung beeinflusst. Bei der Gegenüberstellung der Iterationsschrittzahl $\varnothing n_{\text{it}}$ und der Rechenzeit t_{loop} in 9.15 c) und d) fällt auf, dass der Anteil der Gesamtlöserzeit t_{LGS} durch die geringe Anzahl von Iterationsschritten nicht mehr so stark von der Berechnungsphase abhängt, sondern die Konfigurationsphase erhöhten Einfluss ausübt. Was die zugrundeliegende Gitterknotenanzahl betrifft, so trumps die RCM-Permutation mit einer geringeren Konfigurations- und Berechnungszeit auf. Der Anteil $t_{\text{LGS}}/t_{\text{loop}}$ entspricht am Beispiel ILU(2) bei einer RCM-Vorsortierung 54.0 Prozent, während COLAMD und SFC bei 60.7 und 60.2 Prozent liegen. Wird in den folgenden Untersuchungen nicht explizit die Knotenpermutation erwähnt, so sind die Gitterknoten mittels RCM-Permutation vorsortiert und die Gitterelemente entsprechend nummeriert.

Beim Vergleich der globalen Vorkonditionierungsmethoden werden die Größen $\varnothing t_{\text{konf}}$, $\varnothing t_{\text{it}}$ und $\varnothing n_{\text{it}}$ wieder getrennt betrachtet, siehe Abbildung 9.16. In der Konfigurationsphase ist Schur+GMRES aus denselben Gründen wie im vorherigen Beispiel schneller als die RAS-Vorkonditionierung von Gesamt- und Schursystem. Durch den Datenaustausch wird in diesen Fällen mit $p = 2$ sogar mehr Zeit benötigt, als bei der seriellen Berechnung (bei der keine Konfiguration eines globalen Vorkonditionierers

9. Effiziente Berechnungsmethoden

stattfindet). Die durchschnittliche Berechnungszeit eines Iterationsschritts ist jedoch unabhängig von der parallelen Vorkonditionierung. Dies liegt daran, dass das Abbruchkriterium des GMRES-Verfahrens im Vorkonditionierer schnell erfüllt wird und die Anzahl der kollektiven Kommunikationsaufrufe somit sehr gering ist. Bei der Betrachtung von $\varnothing n_{it}$ fällt auf, dass die Anzahl der benötigten Iterationsschritte in diesem Beispiel unabhängig von der verwendeten parallelen Vorkonditionierungsmethode konstant gering bleibt und somit auch bei anwachsendem p eine beständige Qualität erreicht wird¹. Da Schur+GMRES bei kürzerer Konfigurationszeit einen gleichwertigen Präkonditionierer stellt, ist in diesem Fall diese parallele Vorkonditionierung den RAS-Ansätzen vorzuziehen. Für die ILUT-Faktorisierung dauert bei diesem Ansatz zwar die Konfigurationszeit länger, jedoch wird für die Berechnungsphase weniger Zeit beansprucht.

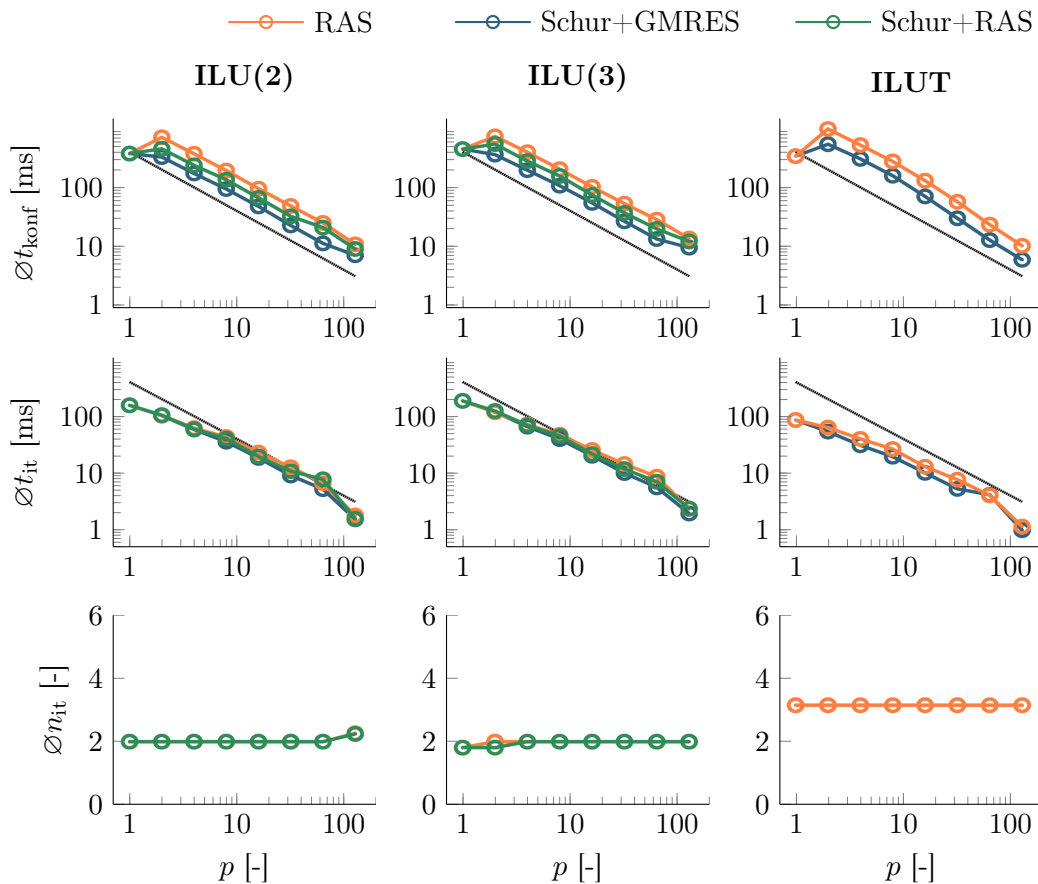


Abbildung 9.16.: Die gemittelten Größen $\varnothing t_{konf}$, $\varnothing t_{it}$ und $\varnothing n_{it}$ bei verschiedenen Kombinationen von paralleler und sequentieller Vorkonditionierung. Die schwarze Linie zeigt als Referenz die ideale Steigung.

¹ Vor Veränderung der Diskretisierung, siehe Titelblattrückseite, konnte die beständige Qualität nur mit einer Schur+GMRES-Vorkonditionierung erreicht werden. Zudem waren insgesamt mehr Iterationsschritte nötig.

Für einen besseren Vergleich ist in Abbildung 9.17 für die Schur+GMRES-Methode die Gesamtlöserzeit mit entsprechender Beschleunigung und Effizienz aufgezeichnet. Die hohen Kosten bei der Konfiguration der ILUT-Zerlegung können aufgrund der geringen Iterationsanzahl nicht durch eine schnelle Berechnung der Iterationsphase kompensiert werden.

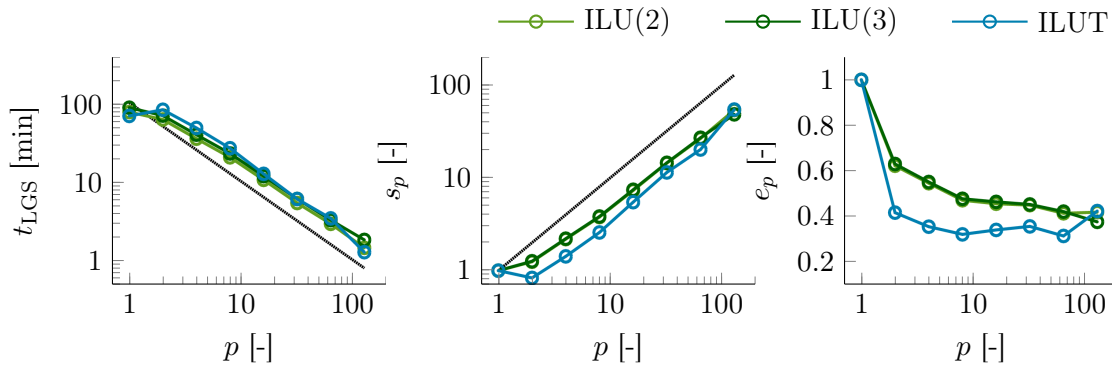


Abbildung 9.17.: Gesamtlöserzeit t_{LGS} , sowie Beschleunigung s_p und Effizienz e_p verschiedener Faktorisierungen bei einer parallelen Schur+GMRES Vorkonditionierung. Die schwarze Linie zeigt die ideale Steigung.

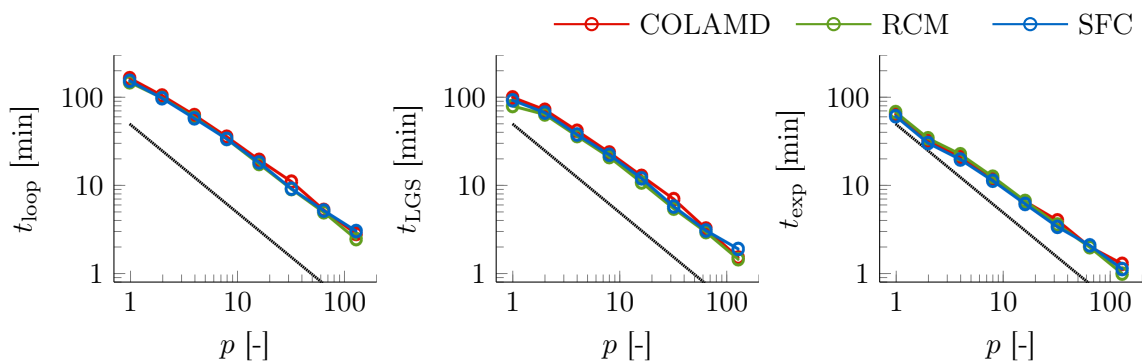


Abbildung 9.18.: Die Rechenzeiten t_{loop} , t_{LGS} und t_{exp} bei verschiedenen Gitternummerierungen mit einer Schur+GMRES/ILU(2)-Vorkonditionierung. Die schwarze Linie zeigt die ideale Steigung.

In Abbildung 9.18 sind am Beispiel einer parallelen Schur+GMRES Vorkonditionierung mit ILU(2)-Faktorisierung für die verschiedenen Gittervorsortierungen die Rechenzeit t_{loop} und deren Bestandteile t_{LGS} und t_{exp} aufgezeichnet. Bei wachsender Prozessanzahl kann die Wahl der Vorsortierung nur einen sehr geringen Einfluss verzeichnen. Beim Lösen des Gleichungssystems wird auch hier von einer RCM-Sortierung profitiert. Diese Vorsortierung wird für alle weiteren Berechnungen verwendet².

² Da mit veränderter Diskretisierung der Lösungsalgorithmus sehr schnell konvergiert, wird t_{loop} nicht vorwiegend von der Gesamtlöserzeit t_{LGS} dominiert, so dass hier auch die SFC-Sortierung eine gute Alternative darstellt.

Insgesamt fällt auf, dass der Anteil der expliziten Berechnung mit zunehmender Prozessoranzahl immer mehr zunimmt. Bei der RCM-Vorsortierung steigt der Anteil $t_{\text{exp}}/t_{\text{loop}}$ beispielsweise von 0.35 für $p = 2$ auf 0.41 bei 128 Prozessoreinheiten. Dies liegt vor allem daran, dass es bei trockenen Knoten außerhalb des variablen Rechengebiets Ω_v nichts zu berechnen gibt. Je mehr Partitionen, desto ungleichmäßiger sind trockene und nasse Knoten verteilt, was einem Ungleichgewicht der Rechenlast auch außerhalb des Gleichungssystems gleichkommt. Diese Problematik wird in Abschnitt 9.4 ausführlicher behandelt.

9.3. Beschränkung auf das Teilsystem

Etwa 80 Prozent der Gitterknoten des für das Mentawai-Szenario verwendete Rechengitters sind zu Beginn trockene Landknoten. Während des Rechenlaufs wird nur ein kleiner Teil dieser Knoten überflutet, so dass sie in die Knotenmenge $\mathcal{N}_w \subset \mathcal{N}_v$ der nassen Knoten aufgenommen werden. Diese Erkenntnis motiviert den Ansatz, sich bei der Berechnung des Bodendrucks auf das variable Rechengebiet Ω_v zu beschränken, das die nassen Knoten mitsamt der direkt angrenzenden trockenen Knoten umfasst, siehe Abschnitt 5.3.

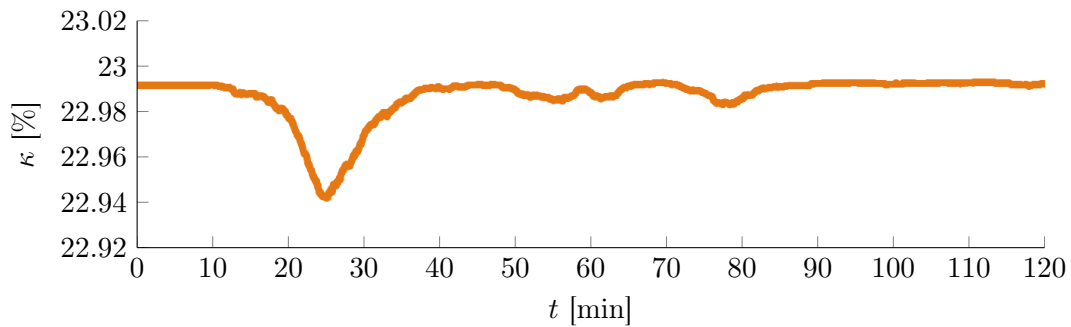


Abbildung 9.19.: Verhältnis $\kappa = |\mathcal{N}_v|/|\mathcal{N}|$ in Prozent.

Jeder Gitterknoten gilt entweder als nass oder als trocken. Das variable Rechengebiet verändert sich, sobald ein Knoten diesen Zustand ändert. In Abbildung 9.19 ist die Veränderung des Rechengebiets über die Zeit aufgetragen³. Das Verhältnis κ beschreibt den Anteil den das variable Rechengebiet Ω_v im Gesamtgebiet Ω einnimmt, gemessen an der Gitterknotenanzahl. Da das Gleichungssystem über die Gitterknoten definiert ist, beschreibt κ den Anteil der Gleichungen, die im reduzierten Teilsystem aus Abschnitt 5.3.2 enthalten sind.

Der Vergleich zwischen der Berechnung des reduzierten Teil- und des Gesamtsystems bei den seriellen Läufen des Mentawai-Szenarios wird in Abbildung 9.20 für die Zeiten $\emptyset t_{\text{konf}}$, $\emptyset t_{\text{it}}$ und t_{LGS} sowie den Füllfaktor f_{LU} gezeigt. Dass im reduzierten System

³ TsunAWI-NH reagiert in der überarbeiteten Version, siehe Titelblattrückseite, nicht mehr so sensibel auf steile Bathymetriegradien. Dies wirkt sich auch auf den Verlauf in Abb. 9.19 aus.

nur ein Teil der Gleichungen eingeht, macht sich vor allem in der durchschnittlichen Berechnungszeit eines Iterationsschritts $\varnothing t_{it}$ bemerkbar. Die Anzahl der benötigten Iterationsschritte n_{it} ist unabhängig davon, ob das Gesamt- oder das reduzierte Teilsystem gelöst wird. Die Komponenten des Lösungsvektors, die im Teilsystem unberücksichtigt bleiben, stehen im Gesamtsystem von vornherein fest und müssen nicht iterativ angenähert werden. Somit wird für das Teilsystem nur ein kleiner Teil der Berechnungszeit beansprucht. Die Konfigurationszeit t_{konf} wird trotz Dimensionsunterschied nicht ganz so stark reduziert, da weder für die Matrix \mathbf{A}_v , noch für die Dreiecksmatrizen \mathbf{L} und \mathbf{U} bei einer $ILU(k)$ -Faktorisierung die Datenstrukturen wiederverwendet werden können. Zudem wird in dieser Phase die aktuelle Indizierung im Teilsystem bestimmt, damit die Matrixeinträge richtig gesetzt werden können. Im Vergleich der Gesamtlöserzeit wird für die Teilsysteme etwa die Hälfte der Zeit beansprucht, die beim Lösen des Gesamtsystems verwendet wird. Der deutliche Anstieg des Füllfaktors f_{LU} bei der ILUT-Zerlegung zeigt, dass im reduzierten System nicht so viele Werte unterdrückt werden wie im Gesamtsystem.

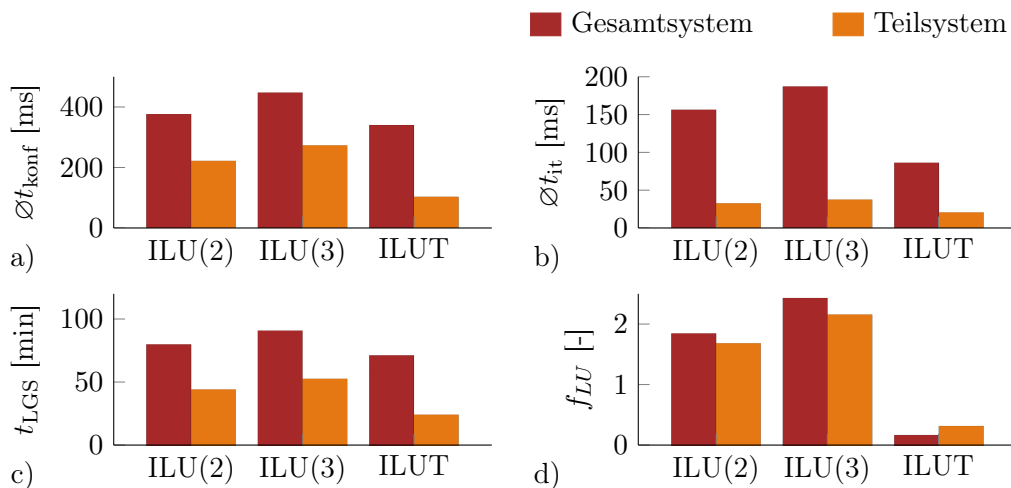


Abbildung 9.20.: a) Konfigurationszeit $\varnothing t_{konf}$, b) Iterationszeit $\varnothing t_{it}$, c) Gesamtlöserzeit t_{LGS} und d) Füllfaktor f_{LU} bei serieller Berechnung des Gesamt- beziehungsweise des reduzierten Teilsystems.

Für die parallelen Läufe werden dieselben Partitionierungen genutzt, die schon bei der Berechnung des Gesamtsystems verwendet wurden. Das Gleichungssystem wird in jedem Zeitschritt aus den pro Partition lokal enthaltenen Teilgebieten von Ω_v zusammengesetzt, wobei kollektive Kommunikationsroutinen verwendet werden. Abbildung 9.21 zeigt die gemittelten Kenngrößen $\varnothing t_{konf}$ und $\varnothing t_{it}$ bei ansteigender Prozessoranzahl p bei einer Schur+GMRES-Vorkonditionierung. Die starken Zeiteinsparungen in Konfigurations- und Anwendungsphase, die bei der seriellen Berechnung durch die Beschränkung auf das reduzierte Teilsystem zu verzeichnen sind, skalieren leider nicht. Die Konfigurationszeit des Löser wird nicht signifikant beschleunigt; die Definition des aktuellen reduzierten Teilsystems und das Aufsetzen der Matrix \mathbf{A}_v wird immer aufwendiger, je mehr Partitionen beteiligt sind. Die entsprechenden Kurven knicken ab und bilden sogar ein Minimum. Das gleiche Verhalten ist auch bei der durchschnitt-

9. Effiziente Berechnungsmethoden

lichen Iterationszeit $\varnothing t_{it}$ zu erkennen. Grund dafür ist die ungleichmäßige Verteilung des variablen Rechengebiets und damit des reduzierten Gleichungssystems.

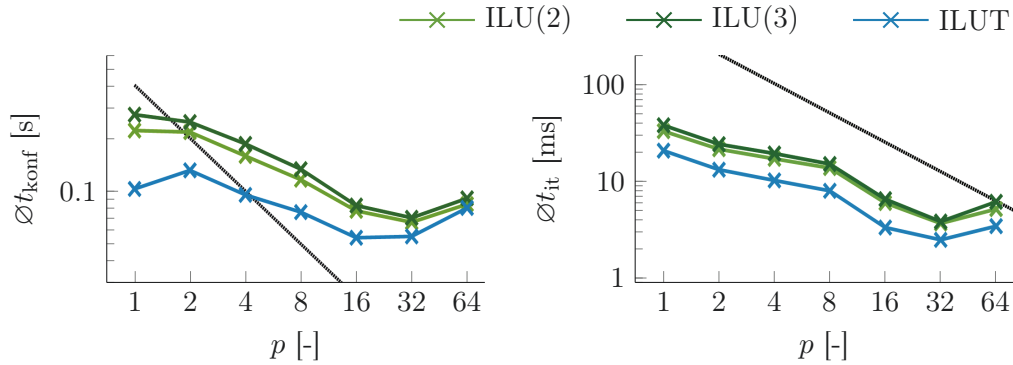


Abbildung 9.21.: Die gemittelten Zeiten für Konfiguration $\varnothing t_{konf}$ und die Berechnung eines Iterationsschritts $\varnothing t_{it}$ beim Lösen des reduzierten Teilsystems mit einer Schur+GMRES-Vorkonditionierung.

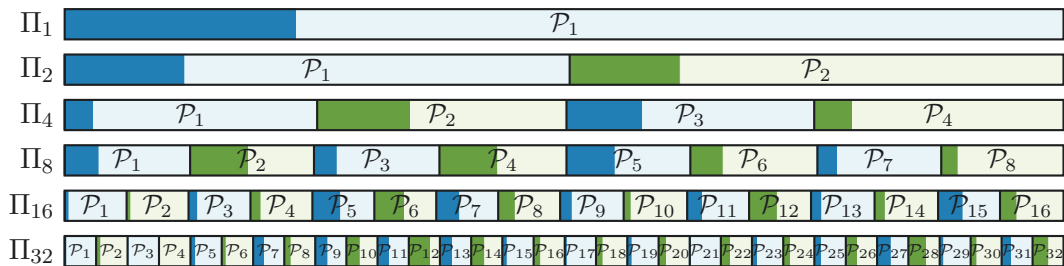


Abbildung 9.22.: Der dunkel eingefärbte Anteil der Knotenmenge \mathcal{N}_v^0 wird ungleichmäßig auf die einzelnen Partitionen verteilt.

In Abbildung 9.22 sind die Verteilungen der Knotenmengen \mathcal{N} und \mathcal{N}_v^0 für die Zerlegungen bis $p = 32$ skizziert. Bei der Berücksichtigung der Verteilung der Gitterknoten des variablen Rechengebiets wird das Lastungleichgewicht beim Lösen des reduzierten Gleichungssystems deutlich. Bei der Zerlegung in zweiunddreißig Teilgebiete gibt es beispielsweise eine Partition, die zu Beginn keinen einzigen Knoten des variablen Rechengebiets enthält.

Der enorme Zeitunterschied zwischen Konfigurations- und Anwendungszeit zusammen mit der geringen Iterationsanzahl führt dazu, dass sich $\varnothing t_{konf}$ stark im Verlauf von t_{LGS} abzeichnet, siehe Abbildung 9.23. Beschleunigung und Effizienz unterstreichen, dass dieser Ansatz bei $p > 16$ unbrauchbar ist. Für eine kleine Anzahl an Prozessoren bildet er jedoch eine durchaus nützliche Alternative zur Berechnung des Gesamtsystems. In Bezug auf die unvollständigen Faktorisierungsansätze ist zu erwähnen, dass die ILUT-Zerlegung beim Lösen des reduzierten Teilsystems effizienter ist als ILU(2). Nicht mehr benachteiligt, da die levelbasierten Zerlegungen ebenfalls die Indexmenge S berechnen und die Datenstrukturen neu anlegen müssen, zeigt dieser Ansatz seine Vorzüge.

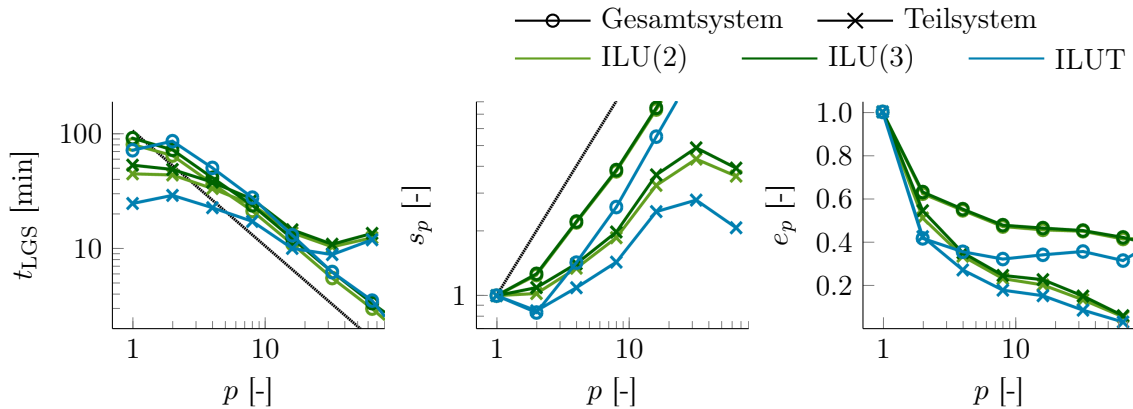


Abbildung 9.23.: Die Gesamtlösungszeit t_{LGS} , Beschleunigung s_p und Effizienz e_p beim Lösen des reduzierten Teil- beziehungsweise des Gesamtsystems mit einer Schur+GMRES-Vorkonditionierung.

9.4. Partitionierung

Bei einer einfachen, ungewichteten Zerlegung wird jeder Partition in etwa dieselbe Anzahl an Knoten zugeordnet, so dass eine gleichmäßige Verteilung erfolgt. Dieses Vorgehen empfiehlt sich, wenn unabhängig vom Gitterknoten dieselben Rechenoperationen durchgeführt werden. Existieren trockene Knoten im Rechengitter, so ist im explizit berechneten Anteil der Tsunamisimulation die Rechenlast heterogen: Da innerhalb eines Zeitschritts nur Gitterknoten des variablen Rechengebiets Ω_v überflutet werden können, wird an allen übrigen (trockenen) Knoten praktisch nichts gerechnet. Beim Lösen des reduzierten Teilsystems gehen ebenfalls nur die Werte bezüglich der Knotenmenge \mathcal{N}_v in die Berechnung des nichthydrostatischen Bodendrucks mit ein. Wird jedoch das Gesamtsystem gelöst, so werden trockene und nasse Knoten gleichbehandelt. Auch wenn die Lösung bezüglich der trockenen Knoten schon bekannt ist, so werden doch dieselben Algorithmen durchlaufen.

9.4.1. Gewichtete Partitionierung

Ist der Rechenaufwand nicht an allen Knoten gleich, so können die Knoten vor der Partitionierung gewichtet werden. Jeder Partition wird daraufhin eine Knotenmenge zugeteilt, deren Gewichte aufsummiert denselben Betrag ergeben. Bei einer Gewichtung im Verhältnis $w_{\text{nass}} : w_{\text{trocken}} = 4 : 1$ kommen zum Beispiel vier trockene Knoten auf einen nassen Knoten. Da das Rechengebiet vor dem Rechenlauf zerlegt wird, kann nur der Status zum Zeitpunkt $t = 0$ in die Gewichtung eingehen, da unklar ist, welche Knoten überhaupt überflutet werden.

Da eine gewichtete Partitionierung keine gleichmäßige Zerlegung der Knotenmenge \mathcal{N} erzeugt, ist dieser Ansatz beim Lösen des Gesamtsystems nicht sinnvoll. Das Lösen des Gleichungssystems beansprucht den größten Rechenanteil bei der nichthydrostatischen Tsunamisimulation und so dominiert die ausgewogene Verteilung der Gesamt-

9. Effiziente Berechnungsmethoden

knotenmenge. Doch auch wenn jeweils nur das reduzierte Teilsystem gelöst wird, erweist sich dieser Ansatz als problematisch. Es ist schwer, eine optimale Gewichtung zu finden. Je größer der Unterschied zwischen w_{nass} und w_{trocken} ist, desto mehr schwankt die Verteilung der Gesamtknotenmenge \mathcal{N} bei kleinen Unterschieden in der Verteilung von \mathcal{N}_v , siehe Abbildung 9.24. Werden bei einer Partition viele von vergleichsweise wenigen Landknoten überflutet, so wird das initiale Gleichgewicht sehr stark gestört. Die entsprechende PE muss Werte für schon immer nasse und überflutete Knoten berechnen, während andere Prozessoren, für wenige nasse, dafür sehr viele trockenen Knoten zuständig sind. Hinzu kommt, dass auch bei einer gewichteten Partitionierung die Verteilung der Knotenmenge \mathcal{N}_v nicht ausbalanciert ist.

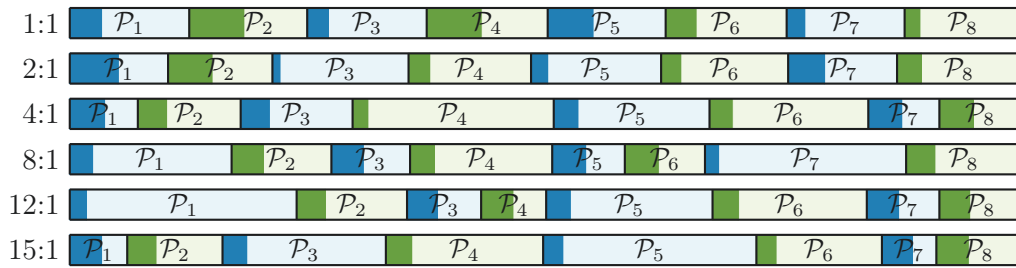


Abbildung 9.24.: Zerlegung der Gitterknoten in acht Partitionen bei unterschiedlicher Knotengewichtung $w_{\text{nass}} : w_{\text{trocken}}$.

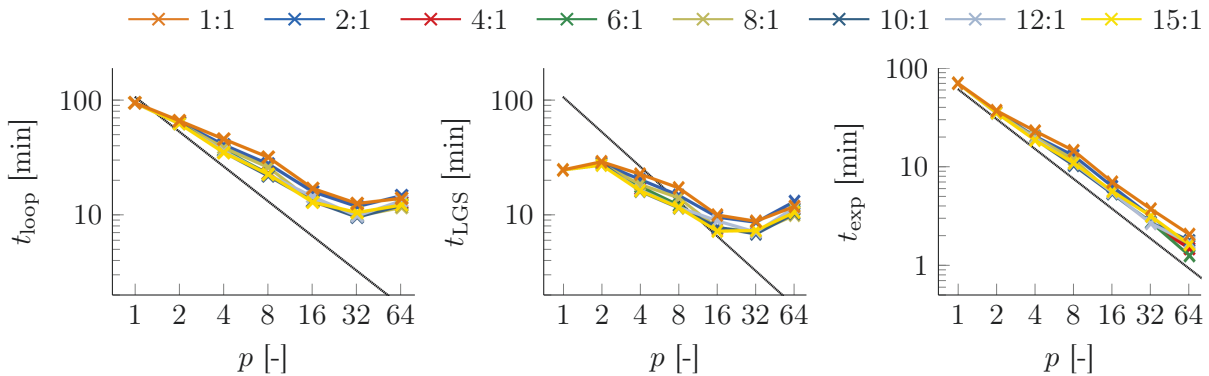


Abbildung 9.25.: Die Rechenzeit t_{loop} und deren Anteile t_{LGS} und t_{exp} mit einer Schur+GMRES/ILUT-Vorkonditionierung bei unterschiedlicher Gewichtung $w_{\text{nass}} : w_{\text{trocken}}$.

Dessen ungeachtet, kann sowohl die Lösung des reduzierten Gleichungssystems und der explizit berechnete Anteil der Tsunamisimulation durch eine gewichtete Partitionierung beschleunigt werden, wie Abbildung 9.25 zeigt. Mit jeder der hier verwendeten Gewichtungen kann die Gesamtlösungszeit t_{LGS} für $2 \leq p \leq 32$ gesenkt werden, das Minimum bei $p = 32$ bleibt jedoch weiterhin bestehen. Ebenso steigt der Einfluss des expliziten Rechenanteils auf die Rechenzeit t_{loop} bei ansteigendem p nicht mehr so stark. Es ist zu beachten, dass die besten Ergebnisse für t_{LGS} und t_{exp} getrennt betrachtet nicht von derselben (gewichteten) Zerlegung erzielt werden.

Die Gewichtung der Knoten muss nicht unbedingt der Form $w_{\text{nass}} : w_{\text{trocken}}$ entsprechen. Die Knoten könnten auch entsprechend gewichtet werden, je nachdem wie wahrscheinlich die trockenen Knoten überflutet werden, beziehungsweise nasse Knoten trocken fallen. Da jedoch nach wie vor mit diesem Ansatz die Knotenmenge \mathcal{N}_v nicht gleichmäßig verteilt wird und die gewichtete Partitionierung sehr schwer zu kontrollieren ist, wird von einer weiteren Untersuchungen der Gewichtung abgesehen und ein anderer Ansatz besprochen.

9.4.2. Partitionierung in zwei Schritten

Beim Lösen des Gesamtsystems ist eine gleichmäßige Verteilung der Knotenmenge \mathcal{N} von Vorteil. Der explizite Rechenanteil profitiert von einer ausgewogenen Verteilung unter Berücksichtigung der Rechenlast, welche sich bei trockenen und nassen Knoten unterscheidet. Für das Lösen des reduzierten Teilsystems ist eine gleichmäßig Verteilung der Knotenmenge \mathcal{N}_v günstig. Ausgehend von diesen Informationen, wird nun eine Gebieszerlegung generiert, die allen genannten Anforderungen gerecht wird. Da die Veränderung von \mathcal{N}_v nicht im Voraus bestimmt werden kann, muss die Menge \mathcal{N}_v^0 zum Zeitpunkt $t = 0$ genügen.

Angenommen, die Knotenmenge \mathcal{N}_v sei gleichmäßig auf die Partitionen verteilt, dann folgt daraus, dass zu einer optimalen Verteilung der Rechenlast bezüglich nass/trocken auch die übrigen trockenen Knoten gleichmäßig auf die Partitionen verteilt werden müssen. Dies führt wiederum zu einer gleichmäßigen Verteilung der Gesamtknotenmenge \mathcal{N} . Eine solche, mehrfach nützliche Zerlegung kann in zwei Schritten generiert werden: Im ersten Schritt wird ausschließlich die (ungewichtete) Knotenmenge \mathcal{N}_v^0 zerlegt, wie Abbildung 9.26 a) an einem Beispiel mit vier Partitionen zeigt. Für diese Knoten ist im zweiten (ungewichteten) Zerlegungsschritt bei der Verteilung der Menge \mathcal{N} die Partitionszugehörigkeit vordefiniert. Diese Vorgehensweise wird vom Partitionierungsprogramm PaToH unterstützt. Die resultierende Gebietszerlegung ist in Abbildung 9.26 b) zu sehen.

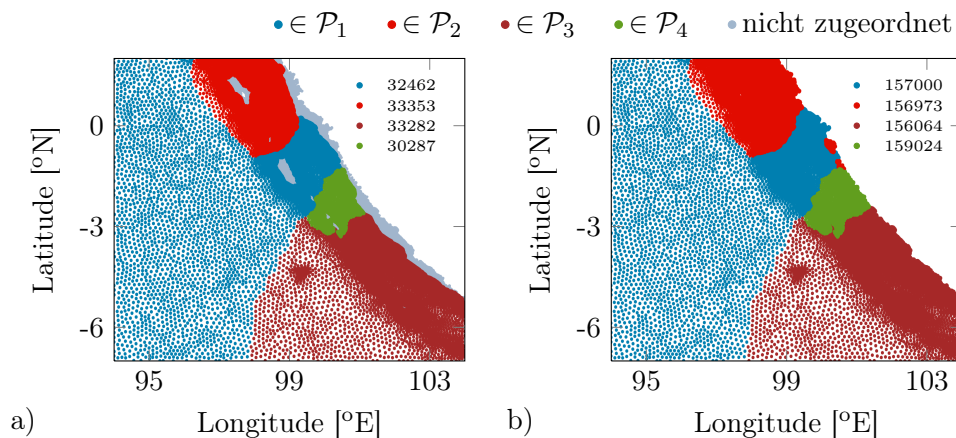


Abbildung 9.26.: a) Die Verteilung der Knotenmenge \mathcal{N}_v^0 und b) die Verteilung der Menge \mathcal{N} inklusive der Knotenanzahl pro Partition.

9. Effiziente Berechnungsmethoden

Bei der zweistufigen Zerlegung kommt es häufig vor, dass die Teilgebiete nicht zusammenhängend sind. Gerade bei der Verteilung der trockenen Knoten treten oft, wie in Abbildung 9.26 b), kleine Partitionsinseln auf. Dabei werden zusätzliche Grenzknoten erzeugt, an denen Daten ausgetauscht werden müssen. Tabelle 9.6 listet auf, welchen Anteil die Grenzknotenmenge $\mathcal{Q} = \bigcup \mathcal{Q}_{ij}$, $i, j \in \{1, \dots, p\}$, in der Knotenmenge \mathcal{N} einnimmt und gleichermaßen den Anteil nasser Grenzknoten $\mathcal{Q}_w^0 = \mathcal{Q} \cap \mathcal{N}_w^0$ in \mathcal{N}_w^0 . Dabei fällt auf, dass bei einer zweistufigen Gebietszerlegung ein viel größerer Anteil der Grenzknoten als nass gilt. Die konstante Qualität der parallelen Vorkonditionierung mit beständig geringer Iterationsanzahl bei wachsender Prozessoranzahl p wird in diesem Fall durch die zweistufige Zerlegung nicht weiter verbessert⁴.

Tabelle 9.6.: Anteil der Grenzknoten innerhalb der Knotenmengen \mathcal{N} und \mathcal{N}_w^0 in Prozent, abhängig von der Partitionsanzahl und Art der Gebietszerlegung.

Zerlegung		Π_2	Π_4	Π_8	Π_{16}	Π_{32}	Π_{64}	Π_{128}
einfach	$ \mathcal{Q} / \mathcal{N} $	0.03	0.18	0.42	0.95	1.58	2.63	4.21
	$ \mathcal{Q}_w^0 / \mathcal{N}_w^0 $	0.01	0.14	0.28	0.79	1.37	2.38	4.02
zweistufig	$ \mathcal{Q} / \mathcal{N} $	0.05	0.90	1.33	2.24	3.16	4.65	6.99
	$ \mathcal{Q}_w^0 / \mathcal{N}_w^0 $	2.15	2.67	2.87	3.36	3.74	4.49	7.25

In Abbildung 9.27 sind die gemittelten Rechenzeiten $\varnothing t_{\text{konf}}$ und $\varnothing t_{\text{it}}$ sowie die Gesamtlösezeit t_{LGS} für eine Schur+GMRES-Vorkonditionierung bei unterschiedlichen unvollständigen LU-Faktorisierungen aufgetragen. Die linke Spalte bezieht sich dabei auf die einfache, ungewichtete Partitionierung, die rechte entspricht den Rechenläufen, denen eine zweistufige Gebietszerlegung zugrunde liegt. Beim Lösen des Gesamtsystems sind für die Berechnung der Iterationsschritte keine Unterschiede festzustellen, die zweistufige Partitionierung wirkt sich jedoch bei wachsendem p positiv auf die Konfiguration der Faktorisierungen im Allgemeinen und der ILUT-Zerlegung im Besonderen aus. Durch die ausgewogene Verteilung trockener und nasser Knoten ist für jede Partition eine vergleichbare Qualität der Faktorisierung sichergestellt. Da jedoch schon bei der einfachen ungewichteten Gebietszerlegung das Gesamtsystem gleichmäßig verteilt wurde, bewirkt hier die zweistufige Partitionierung nur eine geringfügige Verbesserung, verglichen mit den Auswirkungen beim Lösen des reduzierten Teilsystems. Die schlechte Skalierung bei wachsender Prozessorzahl kann zwar nicht gelöst werden, doch für den Einsatzbereich $1 < p \leq 16$ wird durch eine gleichmäßige Verteilung der Knotenmenge \mathcal{N}_v^0 die Berechnung sehr beschleunigt. Dies betrifft sowohl die Konfiguration als auch die Berechnungsphase des Krylov-Unterraumverfahrens.

⁴ Vor der Veränderung der Diskretisierung stieg bei einer zweistufigen Gebietszerlegung die Iterationsanzahl bei wachsendem p langsamer an, als bei einer einfachen ungewichteten Partitionierung.

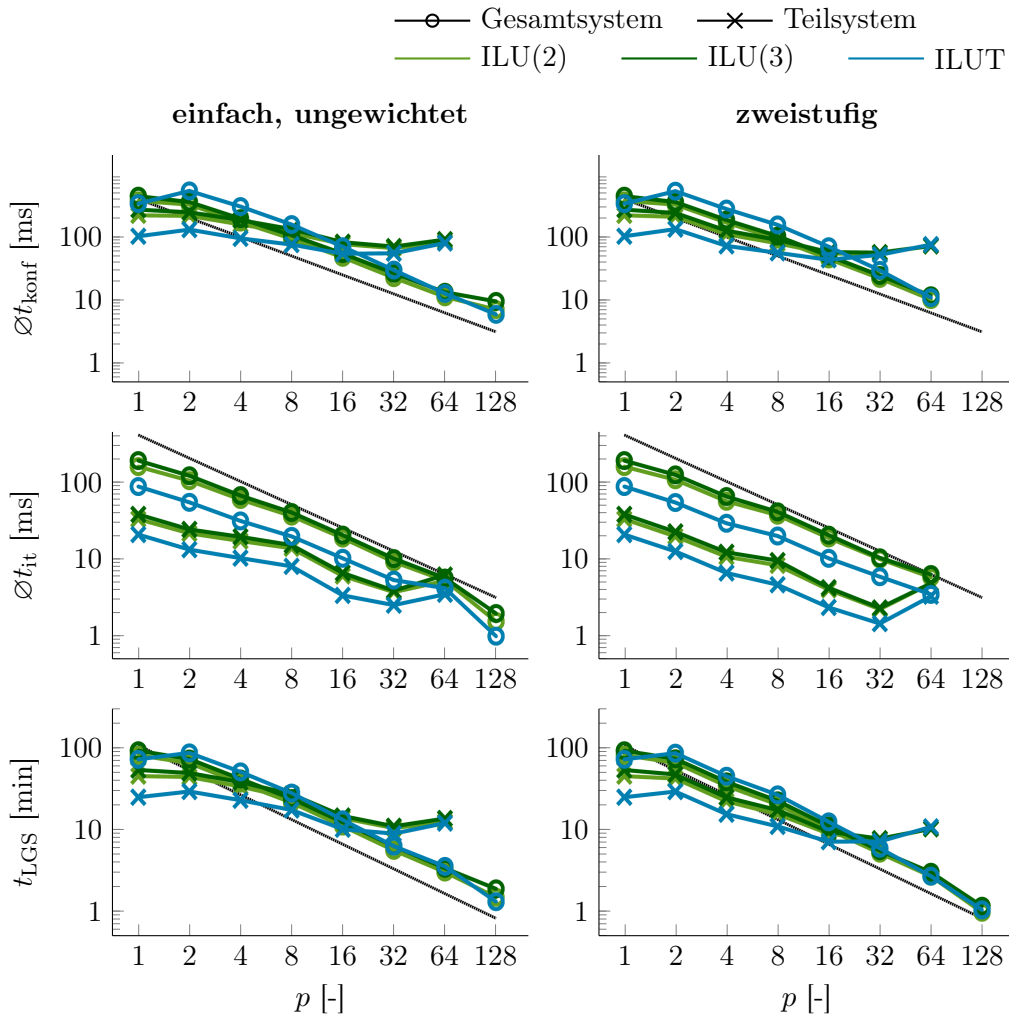


Abbildung 9.27.: Die Rechenzeiten $\varnothing t_{\text{konf}}$, $\varnothing t_{\text{it}}$ und t_{LGS} mit einer parallelen Schur+GMRES-Vorkonditionierung bei einfacher und zweistufiger Gebietszerlegung.

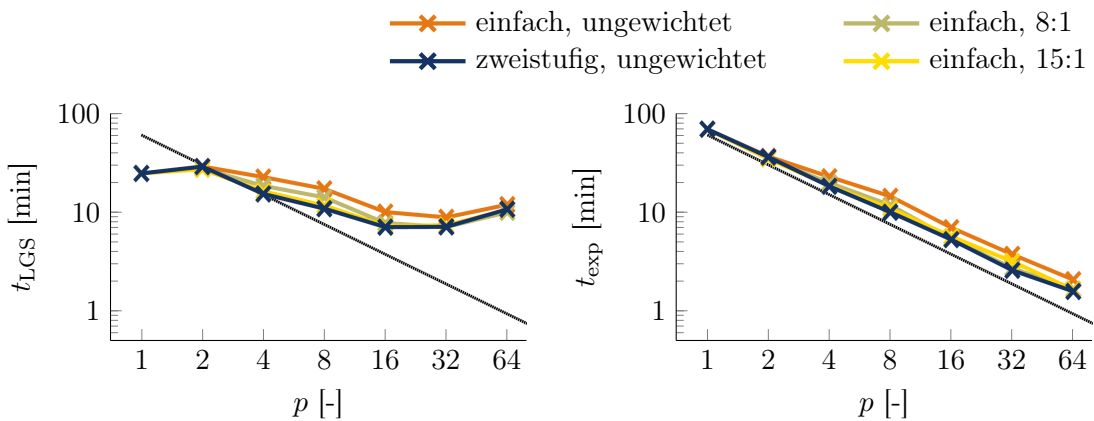


Abbildung 9.28.: Die Rechenzeiten t_{LGS} und t_{exp} mit einer Schur+GMRES/ILUT-Vorkonditionierung bei verschiedenen Gebietszerlegungen.

9. Effiziente Berechnungsmethoden

Im Vergleich der Rechenzeiten t_{LGS} und t_{exp} unter Einbeziehung von gewichteten Partitionierungen in Abbildung 9.28 wirkt sich die zweistufige Gebietszerlegung für $2 < p < 64$ sowohl auf die Gesamtlöserzeit, als auch auf den explizit zu berechnenden Anteil aus. Mit keiner getesteten Gewichtung sind diese Zeiten erreicht worden. Gerade die beschleunigte Berechnung von t_{exp} kommt dabei auch den Rechenläufen zu Gute, bei denen in jedem Zeitschritt das Gesamtsystem gelöst wird. Da die zweistufige Gebietszerlegung im Gegensatz zur gewichteten Partitionierung leicht zu kontrollieren ist und eine schnellere Simulationsberechnung unterstützt, ist dieses Verfahren den alternativ getesteten Methoden vorzuziehen.

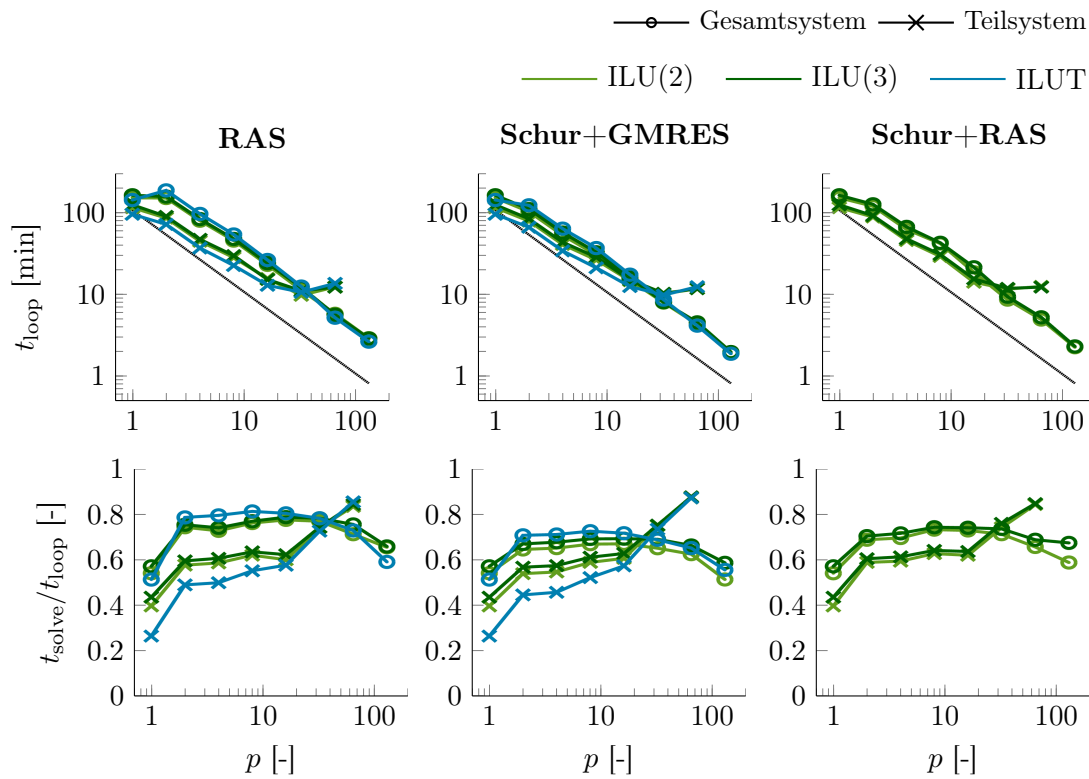


Abbildung 9.29.: Die Rechenzeit t_{loop} und das Verhältnis $t_{\text{solve}}/t_{\text{loop}}$ unterschiedlicher Vorkonditionierungskombinationen für das Gesamt- beziehungsweise das reduzierte Teilsystem bei zweistufiger Gebietszerlegung.

Bei den gemessenen Rechenzeiten t_{loop} und dem zugehörigen Verhältnis $t_{\text{solve}}/t_{\text{loop}}$ für die verschiedenen Vorkonditionierungskombinationen in Abbildung 9.29 wird noch einmal deutlich, dass sich für $p \leq 16$ die Berechnung des reduzierten Teilsystems auszeichnet, da der Löseranteil auf unter 60 Prozent gesenkt werden kann. Da bei $p > 16$ das Verhältnis $t_{\text{solve}}/t_{\text{loop}}$ beim Lösen des reduzierten Teilsystems stark ansteigt, führt hier die Berechnung des Gesamtsystems mit einer Schurkomplement-basierten Vorkonditionierung schneller zum Ergebnis. Bei einer RAS-Vorkonditionierung findet dieser Wechsel erst bei $p = 32$ statt. Bei der Berechnung des reduzierten Teilsystems zeichnet sich die unvollständige LU-Faktorisierung mit dem ILUT-Ansatz aus. Beim Lösen des

Gesamtsystems bildet ILUT erst bei größeren Prozessoranzahlen eine Alternative zu ILU(2). Doch hat die Wahl der parallelen Präkonditionierersmethode mehr Einfluss als die unterschiedlichen Faktorisierungsvarianten. Die Schur+GMRES-Variante besticht hier mit kurzem Konfigurationsaufwand und konstant niedriger Iterationsanzahl bei zunehmender Prozessoranzahl.

10. Simulation von Tsunamiereignissen

In diesem Kapitel werden die Ergebnisse von TsunAWI und TsunAWI-NH bei komplexen Tsunamiszenarien untersucht. Zum einen wird auf einem globalen Rechengitter der Tōhoku-Tsunami vom 11. März 2011 simuliert und mit realen Pegelaufzeichnungen verglichen. Anschließend werden die Ergebnisse des Mentawai-Szenarios verglichen, das im vorherigen Kapitel als Testszenario für technische Untersuchungen verwendet wurde. Als drittes Beispiel wird der Tsunami berechnet, der im Jahr 2003 durch ein Beben vor der algerischen Küste angeregt wurde, wobei verschiedene Anfangsbedingungen verwendet werden. Für alle drei Beispiele werden RCM-basierte Gittervorsortierungen vorgenommen und die Gebietszerlegung mit zweistufiger Partitionierung durchgeführt. In TsunAWI-NH wird zur Vorkonditionierung des Gesamtsystems eine Schur+GMRES/ILU(2)-Kombination verwendet.

10.1. Tōhoku-Szenario

Am 11. März 2011 verursachte ein starkes Beben mit der Magnitude $M_w = 9.0$ und einem Epizentrum an den Koordinaten $(38.297^\circ\text{N}, 142.372^\circ\text{E})$ [69] einen Tsunami der im Nahfeld weite Küstenteile der Region Tōhoku (Honshū, Japan) überflutete und im Fernfeld Orte wie Valparaíso (Chile) oder East Cape (Neuseeland) erreichte. Erdbeben und Tsunami forderten über fünfzehntausend Menschenleben.

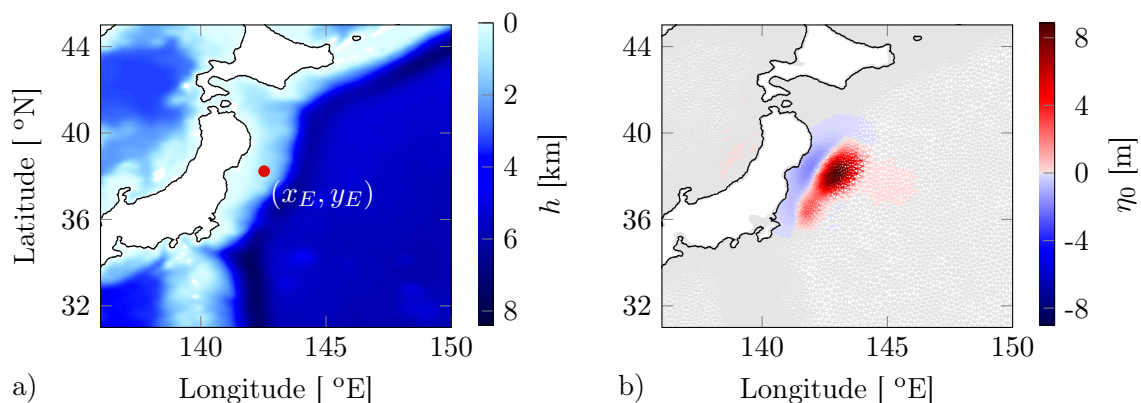


Abbildung 10.1.: a) Die Position des Epizentrums und b) die Auslenkung η_0 [30].

10.1.1. Modellaufbau

Da der Tsunami nicht nur die Ostküste Japans erreichte, sondern sich über den ganzen Pazifik ausbreitete, wird für die Simulation ein globales Gitter verwendet. Die Ränder des Rechengebiets liegen demnach alle an Land. Angeregt wird das Modell durch die Initialauslenkung η_0 [30] bei einer Geschwindigkeit von $\mathbf{u}_0 \equiv 0$ m/s. Abbildung 10.1 zeigt die Lage des Epizentrums und die entsprechende Oberflächenauslenkung zum Zeitpunkt $t = 0$. Es werden die ersten dreißig Stunden nach dem Beben mit einer äquidistanten Zeitschrittweite einer Sekunde berechnet. Die Fakten zur Diskretisierung sind in Tabelle 10.1 zusammengefasst. Mit 64 Prozessoreinheiten benötigt TsunAWI 104 Minuten für die Simulation, TsunAWI-NH mit mehr Rechenaufwand 369 Minuten. In der Gesamtlaufzeit sind hier t_{init} , t_{loop} und t_{out} für die Ausgabe der aktuellen Momentaufnahme alle 120 Zeitschritte eingeschlossen.

Tabelle 10.1.: Diskretisierungsdaten des Tōhoku-Szenarios.

	Parameter	Variable	Größe	Einheit
Raumdiskretisierung	Gittertyp		global	
	Auflösung	Δx	[0.083 39.8]	km
	Knotenanzahl	N	2447367	-
Zeitdiskretisierung	Zeitschrittweite	Δt	1.0	s
	Simulationsdauer	t_{end}	30.0	h
	Zeitschrittanzahl	$n_{\Delta t}$	108000	-

10.1.2. Ankunftszeit und maximale Wellenhöhe

In Abbildung 10.2 sind Ankunftszeit t_a und die maximale Wellenhöhe η_{max} bezüglich der Simulationen mit und ohne nichthydrostatische Korrektur dargestellt.

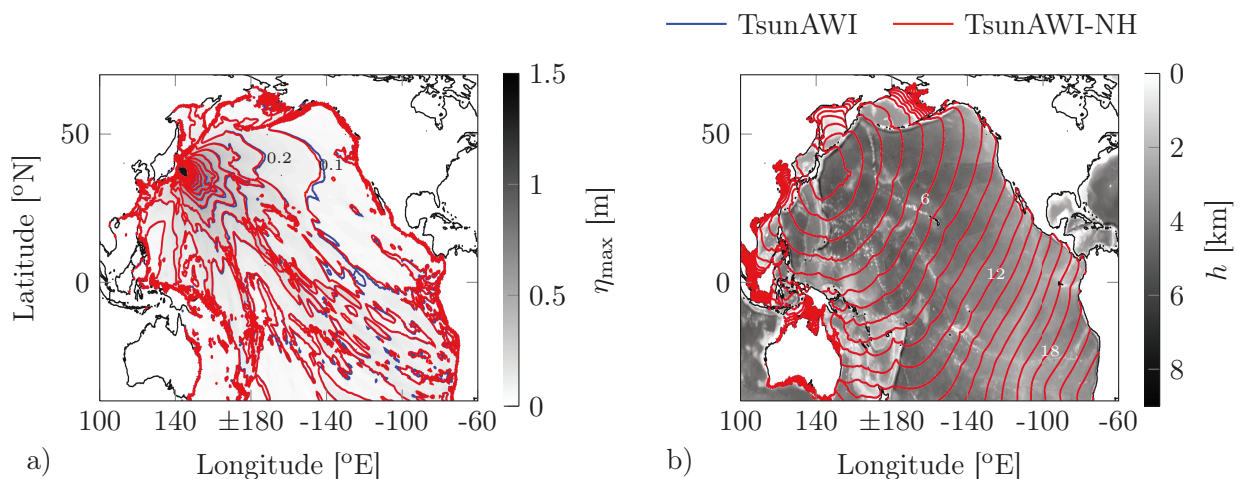


Abbildung 10.2.: Isolinien a) der maximalen Wellenhöhe η_{max} in Dezimeterschritten von 10-150 cm und b) der Ankunftszeit t_a im Stundentakt.

Die beiden Größen $\eta_{\max} : \Omega \rightarrow \mathbb{R}$ und $t_a : \Omega \rightarrow \mathbb{R}$ sind per Definition zeitunabhängig:

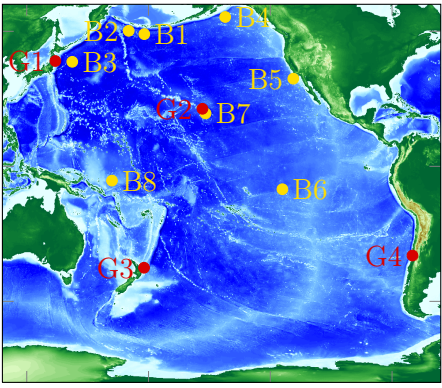
$$\begin{aligned} \eta_{\max}(\mathbf{x}) &:= \max\{\eta(t, \mathbf{x})\}, \\ t_a(\mathbf{x}) &:= \min\{t \in [0, T] : |\eta(t, \mathbf{x})| > 10^{-7}[\text{m}]\}. \end{aligned} \quad (10.1)$$

Zwischen den beiden Modellergebnissen gibt es keine bedeutenden Abweichungen bezüglich der Ankunftszeit. Bei der maximalen Wellenhöhe sind die Differenzen gut sichtbar. Es gibt lokale Unterschiede, die jedoch dieselbe Struktur aufweisen. Bei der Wellenausbreitung wird im nichthydrostatischen Modell die potentielle Energie schneller abgebaut, so dass die Wellenhöhe früher abnimmt.

10.1.3. Pegelstände

Beim Vergleich der Modellergebnisse können aufgezeichnete Pegelstände als Referenz verwendet werden. Da weder das Flachwassermodell noch der nichthydrostatische Ansatz Gezeiten berücksichtigt, wurde der Gezeitenhub aus den Aufzeichnungen herausgefiltert, so dass nur noch die Tsunamiwelle bleibt. Als Referenz werden unter anderem Daten von acht DART[®]-Bojen des National Data Buoy Centers (NDBC) der National Oceanic and Atmospheric Administration (NOAA) [44] genutzt. Zudem gibt es Gezeitenmessungen in Honolulu, Valparaíso, Omaezaki und East Cape, betrieben von den Instituten NOAA (USA), SHOA (Chile), JMA (Japan) und LINZ (Neuseeland). Diese Pegeldata sind über die Sea Level Station Monitoring Facility der Intergovernmental Oceanographic Commission (IOC) [32] verfügbar. Die Position und die Entfernung zum Epizentrum der verschiedenen Stationen sind zusammen mit der Wassertiefe des Referenzknotens im Modell in Tabelle 10.2 zusammengefasst.

Tabelle 10.2.: Die Position der Messstationen, sowie die Distanz zum Epizentrum d und die vorherrschende Wassertiefe h der Referenzknoten im Modell.

ID	Station	x_p [°E]	y_p [°N]	d [km]	h [m]	Position
B1	21414	178.26	48.95	3079.3	5210	
B2	21415	171.85	50.18	2663.3	4680	
B3	21418	148.69	38.71	538.8	5685	
B4	46409	-148.52	55.30	5335.8	4189	
B5	46412	-120.56	32.46	8382.0	3547	
B6	51406	-125.03	-8.48	10801.7	4366	
B7	51407	-156.59	19.59	6161.9	4426	
B8	52406	165.00	-5.29	5366.3	2113	
G1	OMAE	138.23	34.60	557.5	65.8	
G2	HONO	-157.87	21.31	5946.4	158.8	
G3	LOTT	178.16	-37.55	9183.0	80.0	
G4	VALP	-71.63	-33.03	16891.9	59.2	

10. Simulation von Tsunamiereignissen

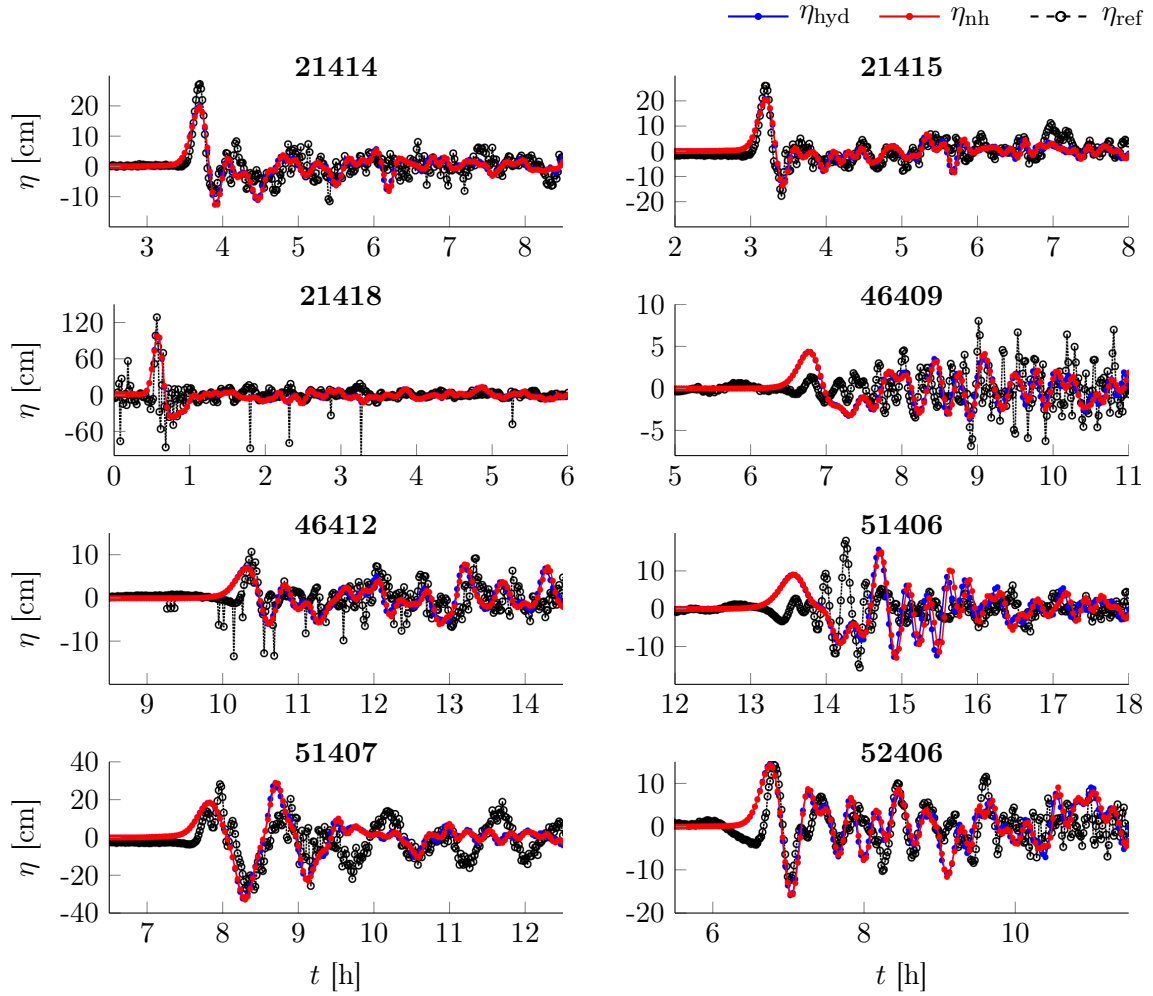


Abbildung 10.3.: Zeitserien der Modellergebnisse mit gefilterten Daten der DART[®]-Bojen als Referenz.

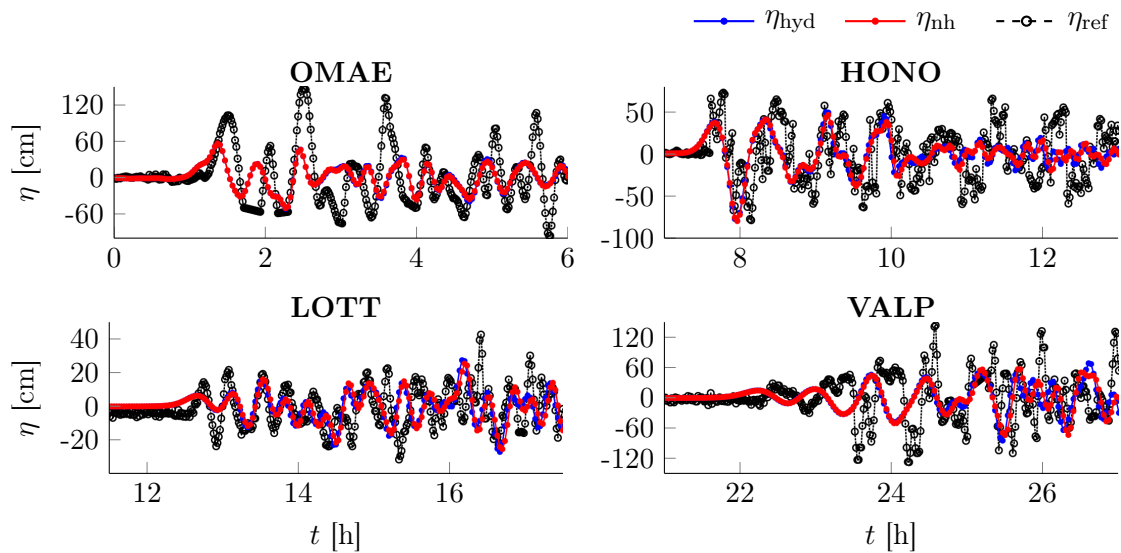


Abbildung 10.4.: Zeitserien der Modelle mit gefilterten Daten der Gezeitenpegel als Referenz.

Die Abbildungen 10.3 und 10.4 zeigen die Modellergebnisse von TsunAWI und TsunAWI-NH im Vergleich mit den gemessenen Referenzdaten der Bojen und Gezeitenpegel. An den Positionen der Bojen herrscht große Wassertiefe, die Gezeitenpegel liefern Vergleichsdaten knapp vor der Küste. Aufgezeigt sind die Zeitserien bezüglich der Gitterknoten, die den Stationskoordinaten am nächsten liegen und $h > 0$ gilt. Die Ankunftszeit der Tsunamiwelle wird von beiden Modellen in den Messstationen nahe des Epizentrums gut abgebildet. Die Übereinstimmung des führenden Wellenbergs deckt sich mit der Ankunftszeit t_a im vorherigen Abschnitt. Bei Stationen, die erst nach dem Zurücklegen einer großen Distanz erreicht werden, kommt die Welle tendenziell zu früh an. Für die ersten Wellenberge stimmt die Phase der Modellrechnungen sowohl bei Bojen als auch bei Gezeitenpegeln oft mit den gemessenen Daten überein. Die Station LOTT auf Neuseeland ist hierfür trotz großer Entfernung zum Epizentrum ein gutes Beispiel. An den Gezeitenpegeln wird die Amplitude der Wellen eher unterschätzt. Trotz einer relativ hohen Auflösung in der Umgebung der Pegel ist die tatsächliche Umgebung nur schlecht dargestellt. Unterschiede zwischen den Modellergebnissen sind vorhanden, doch sehr klein. Während der führende Wellenberg von beiden Modellen gleich berechnet wird, liefern die Modelle unterschiedliche Ergebnisse bei den folgenden Wellenbergen, wenn Reflexionen und Überlagerungen die Gestalt der lokalen Oberflächenauslenkung beeinflussen. Diese Unterschiede werden umso deutlicher, je weiter die Messstation vom Epizentrum entfernt ist. Es findet eine Phasenverschiebung statt: Die Wellenbewegung wird bei der nichthydrostatischen Berechnung im Vergleich zur Flachwassermethode verzögert. Dies ergibt Sinn, da die Ausbreitungsgeschwindigkeit der Flachwasserwelle c_{sw} eine obere Grenze beschreibt. Besonders deutlich zeigt das der Vergleich an den Positionen der DART[®]-Boje 51406 oder des Gezeitenpegels VALP. In beiden Fällen weichen jedoch die Ergebnisse beider Modelle von den Messdaten ab. Die Wassertiefen an den Koordinaten der Gezeitenpegel sind im Modell tief genug, dass an diesen Positionen keine brechenden Wellen vorkommen. Bezogen auf die Amplitude gibt es zu späterem Zeitpunkt generell kleine Unterschiede zwischen den Modellergebnissen, jedoch erfolgen diese nicht systematisch.

10.2. Mentawai-Szenario

Die Fakten bezüglich des Bebens und der Diskretisierung sind in Abschnitt 9.2.5 beschrieben.

10.2.1. Pegelstandsaufzeichnungen als Referenz

Als Referenzdaten können auch hier Aufzeichnungen verschiedener Messstationen genutzt werden, deren Gezeitenbewegung herausgefiltert wurde. Die im GITEWS-Projekt eingesetzte Boje Sumatra 3 (SU03) [59] lieferte Daten bei einer Meerestiefe von knapp 5600 Metern. Des Weiteren können Gezeitenmessungen der Stationen Telukdalam (TELU) und Tanahbala (TNBL) verwendet werden. Diese Stationen wurden ebenfalls im GITEWS-Projekt eingerichtet. Zusätzlich gibt es eine Station des Sea Le-

10. Simulation von Tsunamireignissen

vel Centers der Universität Hawaii in Padang (PADA). Die Pegel­daten werden über [32] bereit­gestellt. Die vergleichenden Modellergebnisse beziehen sich auf die Werte des den Koordinaten nächsten Gitterknotens mit $h > 0$. Die an den Referenzknoten vorliegende Wassertiefe h ist in Tabelle 10.3 zusammen mit der Position und der Entfernung zum Epizentrum d zu finden.

Tabelle 10.3.: Position der Messstationen, die Entfernung zum Epizentrum d und die Wassertiefe h an den Referenzknoten im Modell.

ID	Station	x_p [°E]	y_p [°N]	d [km]	h [m]	Position
B1	SU03	98.90	-2.80	161.92	5144	
G1	TELU	97.82	0.55	518.44	18.3	
G2	TNBL	98.50	-0.53	376.60	18.3	
G3	PADA	100.38	-1.00	274.27	4.0	

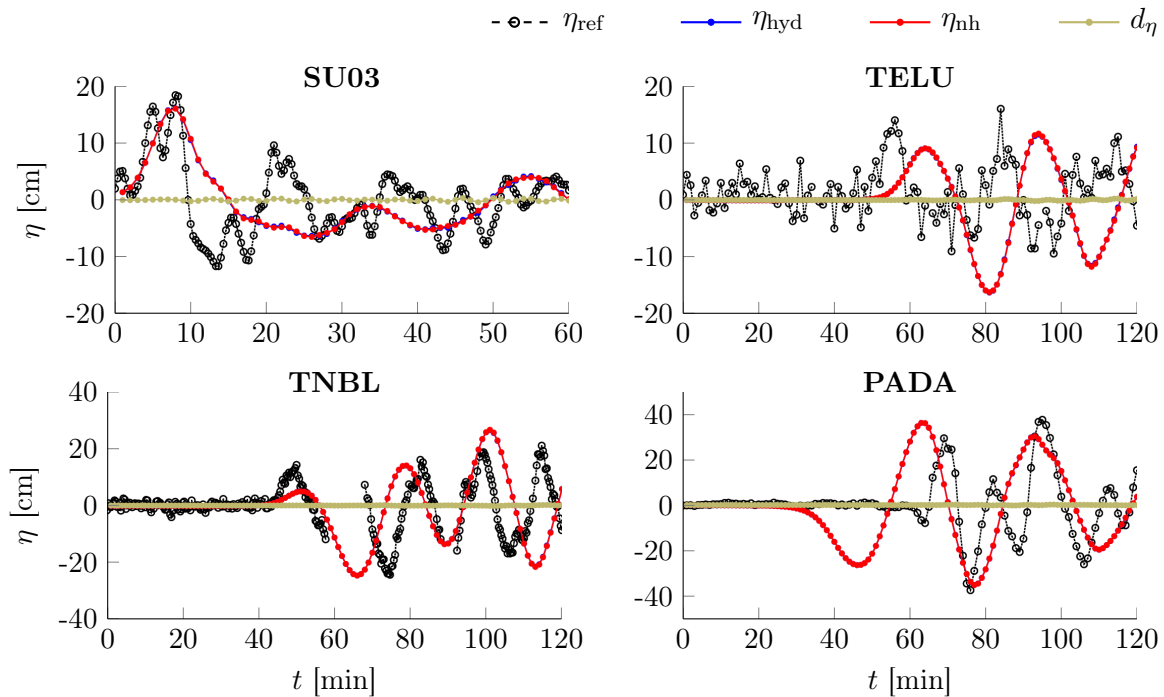


Abbildung 10.5.: Zeitserien bezüglich TsunAWI und TsunAWI-NH mit gemessenen Pegelständen als Referenz, sowie die Distanz $d_\eta = \eta_{\text{hyd}} - \eta_{\text{nh}}$.

In Abbildung 10.5 sind die nachprozessierten Wasserstandsanzeigen der vier Stationen, sowie die entsprechenden Zeitserien der verschiedenen Modellläufe abgebildet. Auf

den ersten Blick wird klar, dass an diesen Positionen keine bedeutenden Unterschiede zwischen den hydrostatischen und nichthydrostatischen Berechnungen zu verzeichnen sind. Die Distanz $d_\eta = \eta_{\text{hyd}} - \eta_{\text{nh}}$ verläuft im Millimeterbereich. Bei den Stationen TELU und TNBL auf der Inselkette kommt die modellierte Welle zu spät an, an der Küstenstation PADA hingegen zu früh. Die nichthydrostatische Modellierung zeigt in dieser Hinsicht keine Verbesserung zum Flachwassermodell.

10.2.2. Ankunftszeit und maximale Wellenhöhe

Ein Vergleich der Ankunftszeit t_a und der maximalen Wellenhöhe η_{max} aus Gleichung (10.1) bezogen auf die Berechnungen von TsunAWI und TsunAWI-NH wird in Abbildung 10.6 mittels Isolinien dargestellt.

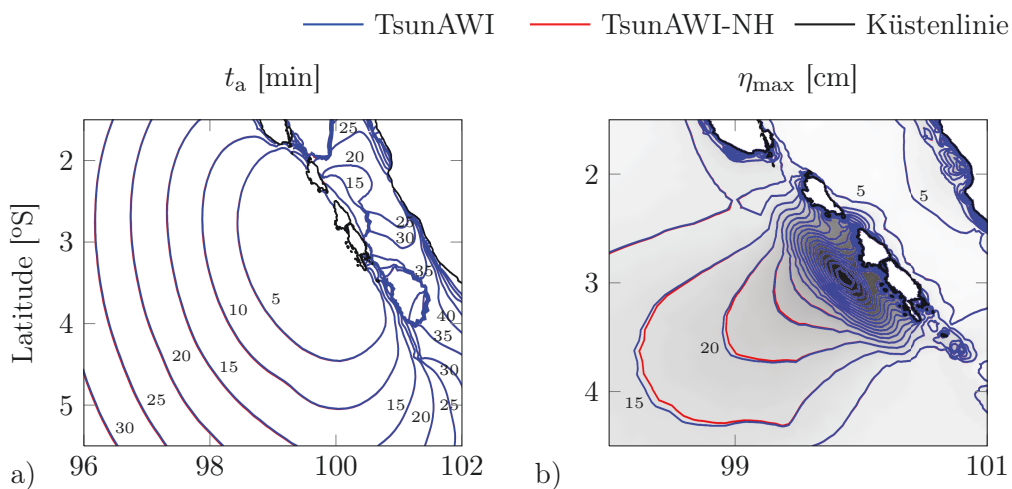


Abbildung 10.6.: Isolinien bezüglich a) der Ankunftszeit t_a beziehungsweise b) der maximalen Wellenhöhe η_{max} im Abstand von 5 cm für 5-100 cm.

Während die nichthydrostatische Korrektur keine signifikante Auswirkung auf die Ankunftszeit t_a hat, sind bei der maximalen Wellenhöhe η_{max} Unterschiede sichtbar. Bei der Wellenausbreitung in die tieferen Gewässer verliert die nichthydrostatische Welle schneller an Höhe, als beim Flachwassermodell. Bei der Ausbreitung in Richtung Küste sind jedoch kaum Unterschiede zu verzeichnen.

10.3. Bourmedès-Zemmouri-Szenario

Nichthydrostatische Effekte treten unter anderem bei submarinen Hindernissen in geringer Wassertiefe auf, wie das Tankexperiment in Abschnitt 4.3 zeigt. Gerade im Mittelmeer mit seiner beckenförmigen Geometrie, siehe Abbildung 10.7, können deutliche Unterschiede der Modellergebnisse erwartet werden. Hier wird ein Tsunami simuliert, der am 23. Mai 2003 im westlichen Mittelmeer durch ein Beben der Magnitude

10. Simulation von Tsunamieignissen

$M_w = 6.8$ angeregt wurde. Das Epizentrum des Bebens lag mit den Koordinaten (36.90 °N, 3.71 °E) [69] vor der Küste Algeriens.

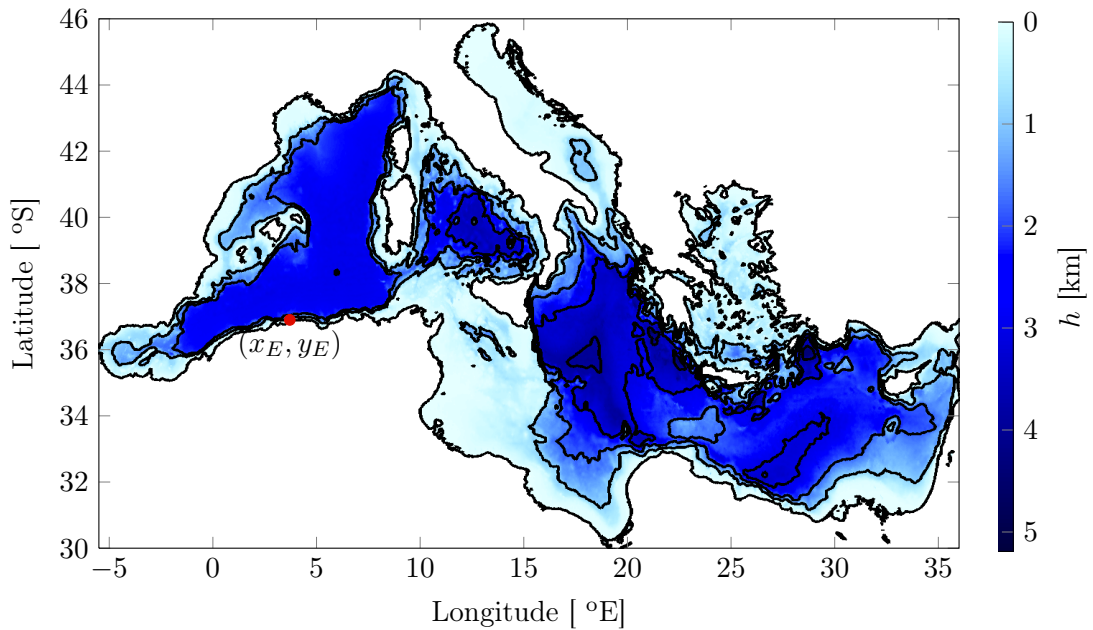


Abbildung 10.7.: Isolinien in 1000 Meter Schritten von 0 bis 4000 Meter Wassertiefe beschreiben die Bathymetrie des Mittelmeeres. Das Epizentrum des Bourmedès-Zemmouri-Bebens liegt direkt vor der Küste Algeriens.

10.3.1. Modellaufbau

Die verwendete Triangulierung \mathcal{T} überdeckt das Mittelmeer mit einer Auflösung von bis zu 29 Metern. Es werden die ersten drei Stunden nach dem Beben simuliert mit einer äquidistanten Zeitschrittweite von 0.5 Sekunden. Die Diskretisierungsdaten sind in Tabelle 10.4 zusammengefasst.

Tabelle 10.4.: Diskretisierungsdaten des Bourmedès-Zemmouri-Szenarios.

	Parameter	Variable	Größe	Einheit
Raumdiskretisierung	Gittertyp		regional	
	Auflösung	Δx	[0.029 10.2]	km
	Knotenanzahl	N	2337783	-
Zeitdiskretisierung	Zeitschrittweite	Δt	0.5	s
	Simulationsdauer	T_{end}	3.0	h
	Zeitschrittzahl	$n_{\Delta t}$	10800	-

Das Szenario wird sowohl mit TsunAWI als auch mit TsunAWI-NH mit den Initialauslenkungen η_0^{O1} und η_0^{O2} und der Anfangsgeschwindigkeit $\mathbf{u}_0 \equiv 0$ angeregt. Die Oberflächenauslenkungen wurden mit Hilfe des Okada-Halbraumverfahrens [46] modelliert. Dafür werden Bruchparameter definiert, welche in Abbildung 10.8 skizziert sind. Für das Szenario wurden zum einen die Parameter des Best-Fit Modells von [74] in die hier verwendeten Bruchparameter umgerechnet (O1). Alternativ wurde zudem die Anregung aus [41] umgesetzt (O2). Im Fall O2 besteht der Bruch aus zehn Plattensegmenten der Länge 4.5 oder 7.5 Kilometer die eine unterschiedliche Auslenkung D erfahren. Die Tiefe T wurde hier für alle Segmente einheitlich auf 1000 Meter gesetzt. Die Bruchparameter für beide Anregungen sind in Tabelle 10.5 zu finden. Die Koordinaten (x_P, y_P) beschreiben dabei die Lage des Punktes P aus Abbildung 10.8. In Abbildung 10.9 ist die berechnete Anfangsauslenkung für beide Ansätze dargestellt.

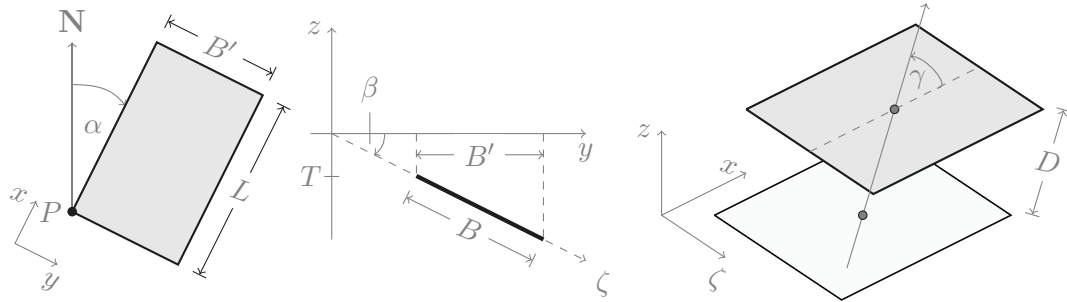


Abbildung 10.8.: Skizze der Bruchparameter. Die horizontalen, kartesischen Achsen x, y werden über den Azimuth α ausgerichtet, β beschreibt den Winkel zur Oberfläche und γ den Bewegungswinkel.

Tabelle 10.5.: Bruchparameter zum Bourmedès-Zemmouri-Szenario: O1 nach [74]; O2 nach [41], wobei die Platte in zehn Segmente aufgeteilt ist.

ID	x_P [°E]	y_P [°N]	L [km]	B [km]	T [km]	α [°]	β [°]	γ [°]	D [cm]
O1	3.37	36.82	32	14	4	65	42	84	180
O2	3.39	36.74	54	15	1	54	50	90	[20,160]

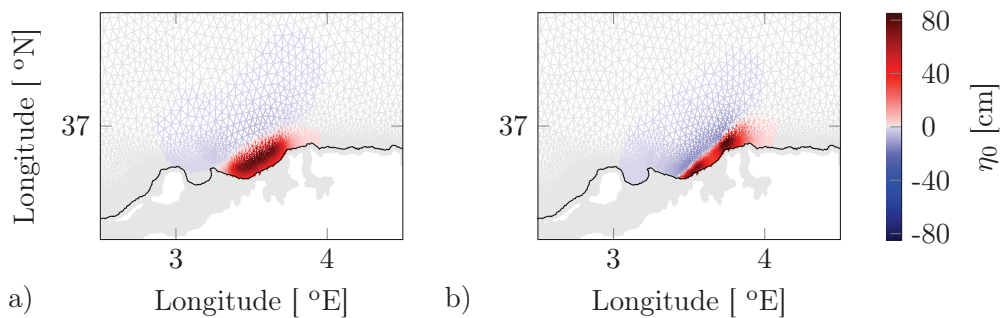


Abbildung 10.9.: Auslenkung η_0 für a) die Anregung O1 und b) die Anregung O2.

10.3.2. Ankunftszeit und maximale Wellenhöhe

In Abbildung 10.10 sind für die zwei Anfangsbedingungen die berechnete maximale Wellenhöhe η_{\max} beider Modelle mittels Isolinien aufgetragen. Da das Epizentrum so nahe vor der algerischen Küste liegt geht ein großer Anteil der freigesetzten Energie aufs Land und nicht auf die Wassermassen über. Die Wellenhöhe η_{\max} nimmt in der Propagationsphase schnell ab. Bei der O2-Anregung generell etwas schneller, als bei der O1-Anregung. Wie schon bei der Berechnung der vorigen Szenarien beobachtet, verliert die nichthydrostatisch modellierte Welle bei der Ausbreitung schneller an Höhe. Der Form der Isolinien bestätigt, dass die maximale Wellenhöhe im nichthydrostatischen Fall stärker von der Bathymetrie beeinflusst wird als beim Flachwassermodell.

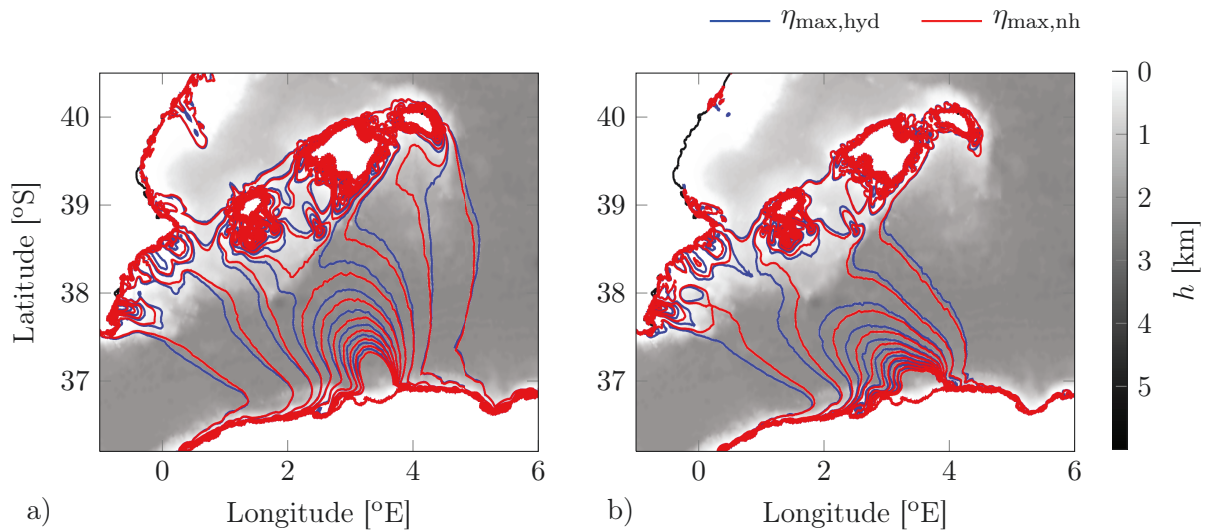
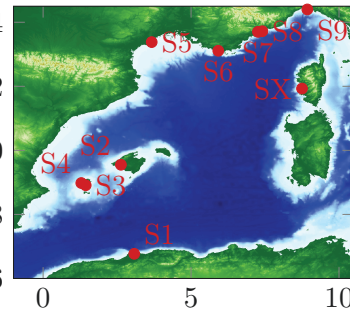


Abbildung 10.10.: Isolinien der maximalen Wellenhöhe η_{\max} im Wertebereich 1-10 cm und einem Intervall von 1 cm bei Anregung a) O1 und b) O2.

Tabelle 10.6.: Position der Häfen und die Distanz zum Epizentrum, sowie die vorherrschende Wassertiefe h des nächstgelegenen nassen Knotens.

ID	Stadt	x_p [°E]	y_p [°N]	d [km]	h [m]	Position
S1	Algier	3.08	36.76	58.2	10.2	
S2	Palma	2.63	39.55	309.4	11.1	44
S3	Ibiza	1.44	38.91	299.3	16.6	42
S4	S. Antoni	1.29	38.98	313.9	4.6	40
S5	Sète	3.67	43.38	720.6	8.8	38
S6	Toulon	5.92	43.11	715.6	1.4	36
S7	Nizza	7.26	43.70	813.6	62.7	
S8	Monaco	7.41	43.72	820.5	103.8	
S9	Genua	8.93	44.40	942.6	18.9	
SX	Ajaccio	8.75	41.93	707.0	13.8	



Statt Gezeitenpegel oder Bojendaten werden hier diskrete Beobachtungsdaten als Referenz genutzt, die sich auf die maximale Wellenhöhe und die Ankunftszeit in den Häfen verschiedener Küstenstädte beschränken. Diese wurden in [58] veröffentlicht, mit dem Hinweis, dass nur die Pegel auf den balearischen Inseln eine ausreichend hohe Abtastrate verfügten, um die maximale Wellenhöhe richtig zu erfassen. Die geschätzte Position der Pegel, sowie die Distanz zum Epizentrum und die vorherrschende Wassertiefe des nächsten nassen Gitterknotens sind in Tabelle 10.6 aufgelistet.

Die maximale Wellenhöhe η_{\max} im Vergleich zu den Werten aus [58] ist in Abbildung 10.11 für die verschiedenen Rechenläufe dargestellt. Nur im Hafen Algiers wird von beiden Anregungen die maximale Wellenhöhe überschätzt. Bei der geringen Abtastrate des dortigen Pegels ist es jedoch möglich, dass auch an dieser Stelle die tatsächliche Welle höher war. Die relativ hohen Auslenkungen auf den Balearen (S2-S4) werden weit unterschritten und die ohnehin geringen Amplituden an der französisch-italienischen Küste werden bei weitem nicht erreicht. Bei der Anregung O2 kommen insgesamt kleinere Wellen in den Häfen an, als bei der O1-Anregung, wie der Vergleich der Isolinien in Abbildung 10.10 erwarten ließ. Unterschiede zwischen dem nichthydrostatischen und dem Flachwassermodell sind vorhanden, jedoch gering.

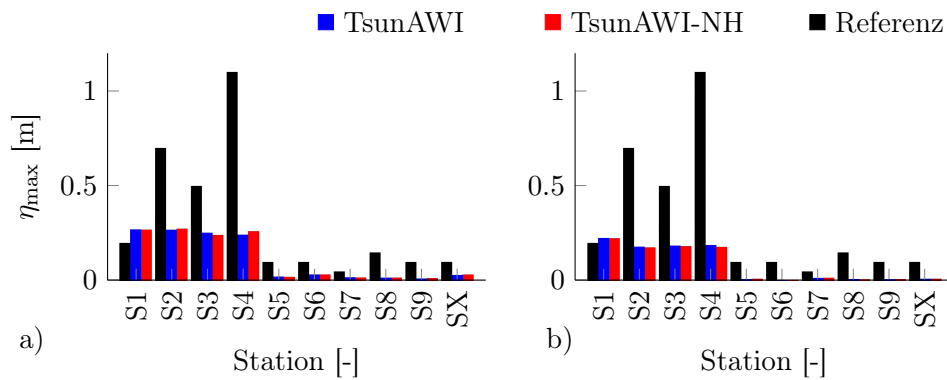


Abbildung 10.11.: Die maximale Wellenhöhe bei den Anregungen a) O1 und b) O2.

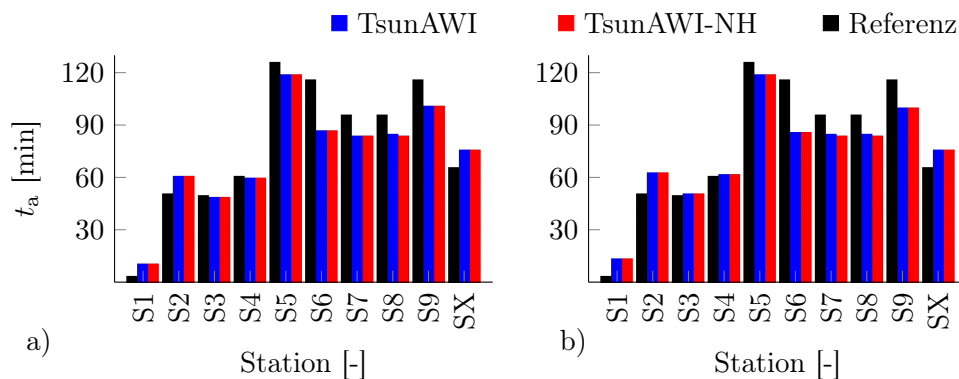


Abbildung 10.12.: Die Ankunftszeit $t_a(\mathbf{x}_p)$ bei den Anregungen a) O1 und b) O2.

10. Simulation von Tsunamiereignissen

Bei der Betrachtung der Ankunftszeit t_a in Abbildung 10.12 wird deutlich, dass beide Modelle mit beiden Anregungen gut beschreiben, wann die Welle die Häfen Ibiza und S. Antoni (S3,S4) erreichte. In Algier, Palma und Ajaccio kommen die berechnenden Wellen etwa 10 Minuten zu spät und in den Häfen der französisch-italenischen Küste bis zu 30 Minuten zu früh an. Jedoch ist die Ankunftszeit t_a^{ref} nicht wohldefiniert. Die Unterschiede zwischen den beiden Modellergebnissen sind minimal.

Die Abweichung zwischen den hydrostatischen und nichthydrostatisch modellierten Ergebnissen ist alles in allem sehr gering. Die Erfahrungen aus den anderen berechneten Tsunamiszenarien legen nahe, dass jedoch die Approximation guter Anfangsbedingungen und Quellterme einen entscheidenden Einfluss auf die Qualität der Lösung hat.

11. Zusammenfassung und Ausblick

Die vorliegende Arbeit beschäftigt sich mit der Untersuchung von verschiedenen numerischen Ansätzen für die effiziente Umsetzung eines nichthydrostatischen Tsunami-modells für komplexe Anwendungen. Für dessen Implementierung wurde das Simulationsprogramm TsunAWI zu TsunAWI-NH erweitert, um die nichthydrostatische Berechnung zu ermöglichen. Die Diskretisierung des Programms TsunAWI basiert auf einer \mathcal{P}^1 - $\mathcal{P}_{\text{NC}}^1$ Finite Elemente Methode auf unstrukturierten Gittern und dem Leapfrog-Zeitschrittverfahren. Die nichtlinearen, hydrostatischen und vertikal gemittelten Flachwassergleichungen, die TsunAWI in seiner Originalversion zugrunde liegen, werden in TsunAWI-NH gemäß der Arbeiten von [12], [62] und [72] als Zwischenlösung verwendet. Die so vorliegende horizontale Geschwindigkeit kann anschließend im nichthydrostatischen Kontext korrigiert werden. Zusätzlich werden hierfür die vertikale Geschwindigkeitskomponente w und der hydrodynamische Bodendruck q bestimmt. Dieser Vorgang wird in jedem Iterationsschritt der Zeitintegration durchgeführt.

Bei den Berechnungen der stehenden Welle im Becken bei unterschiedlicher Beckentiefe und konstanter Wellenlänge zeigt TsunAWI-NH auch dann noch eine gute Übereinstimmung zur analytischen Lösung, wenn das Flachwassermodell schon stark von der Referenz abweicht. Beim Vergleich der Ergebnisse bezogen auf ein Tankexperiment mit linearem Anstieg zur Küste, werden bei einer nichtbrechenden Welle die Ergebnisse von beiden Modellansätzen gut abgebildet. Bei der Berechnung einer brechenden Welle zeigt TsunAWI-NH dann eine deutliche Verbesserung im Vergleich zu TsunAWI. Als Referenzwerte dienen in beiden Fälle Messdaten eines Tankexperiments [63]. Bei einem weiteren Tankexperiment, das die Überströmung eines Unterwasserhindernisses nachstellt, kann das dispersive Zerfallen in zwei Wellenberge von TsunAWI nicht dargestellt werden. Der nichthydrostatische Ansatz liefert hingegen sehr gute Ergebnisse, die bei Messstationen über dem Hindernis eine Korrelation von über 75 Prozent zu den gemessenen Laborwerten aufzeigen. Diese Untersuchungen belegen, dass mit dem hier verwendeten nichthydrostatischen Modell Wellenbewegungen abgebildet werden können, die vom Flachwassermodell nicht erfasst werden.

Motiviert durch die vielversprechenden Ergebnisse bezüglich der eben erwähnten Experimente wurde untersucht, ob auch komplexe Tsunamiszenarien mit dem nichthydrostatischen Modell in vernünftiger Zeit simuliert werden können. Während die Berechnung von w mittels Diagonalisierung explizit ausgeführt wird, muss für die Bestimmung von q ein lineares Gleichungssystem mit unsymmetrischer und zeitabhängiger, doch struktursymmetrischer Systemmatrix gelöst werden. In dieser Arbeit wurden verschiedene Ansätze untersucht, um die Gleichungssysteme effizient zu lösen und somit die nichthydrostatische Simulation zu beschleunigen. Zum Lösen des Gleichungssystems wird das vorkonditionierte Krylov-Unterraumverfahren FGMRES

11. Zusammenfassung und Ausblick

mit Neustart aus dem Löserpaket pARMS 3.2 [38] verwendet, wobei verschiedene Präkonditionierungsmethoden getestet, kombiniert und verglichen wurden. Getestete Routinen die nicht von pARMS standardmäßig unterstützt werden, wurden implementiert und in das Paket eingebettet. Verschiedene Varianten der unvollständigen LU-Zerlegung sowie verschiedene parallele Vorkonditionierungsmethoden wie BJ, RAS und die verschachtelten Ansätze Schur+GMRES und Schur+RAS wurden an zwei Beispielen untersucht, die unterschiedliche Anforderungen an eine effiziente Präkonditionierung stellen. Die Wahl der globalen Präkonditionierungsmethode ist dabei entscheidender als die Wahl der ILU-Variante für das lokale Teilsystem. Bei der Berechnung einer stehenden Welle im Becken, fließt stets im gesamten Rechengebiet der nichthydrostatische Druckterm in die Wellenbewegung mit ein. Beim Lösen des Gleichungssystems ist der Konfigurationsanteil für Matrix und Präkonditionierung sehr gering im Vergleich zum Iterationsanteil. Deshalb wird von einer Vorkonditionierung

- eine gute Annäherung an \mathbf{A}^{-1}
- und algorithmische Effizienz in der Anwendung

verlangt. Bei der parallelen Berechnung konvergieren die Schuransätze mit deutlich weniger Schritten. Schur+RAS ist dabei günstiger in der Anwendung. Bei der komplexen Simulation eines realen Tsunamireignisses verändert sich nur in einem Teil des Rechengebiets der dynamische Bodendruck, ein großer Anteil der Rechenknoten liegt an Land. Mit einer zweistufigen Gebietszerlegung kann eine bessere Verteilung der Rechenlast auf die Prozessoreinheiten erreicht werden. Von den Vorkonditionierungsmethoden wird hier

- eine gute Annäherung an \mathbf{A}^{-1}
- und eine günstige Aufstellungskomplexität in der Konfiguration

gefordert. Schur+GMRES liefert unabhängig von der Prozessoranzahl p eine beständig hohe Qualität mit schneller Konvergenz und ist zudem schnell konfiguriert. Bei vielen Landknoten im Rechengebiet lohnt es sich bei kleinen Prozessoranzahlen das Gesamtsystem auf die Gleichungen zu beschränken, die sich auf die Menge der nassen Knoten und der direkt angrenzenden Landknoten beziehen. Bei wachsender Prozessoranzahl überwiegt jedoch der Kommunikationsaufwand beim Aufstellen der Matrix, so dass für $p \leq 16$ das reduzierte Gleichungssystem und für $p > 16$ das Gesamtsystem schneller berechnet wird. Eine verbesserte Füllstruktur der ILU-Faktorisierungen und eine cache-effiziente Datenspeicherung kann durch die Umsortierung der Gitternummerierung erreicht werden. Eine solche Vorsortierung sollte unbedingt durchgeführt werden. Im Beispiel der stehenden Welle eignet sich eine RCM-Sortierung der Gitterknoten, bei der komplexen Tsunamisimulation sind die Unterschiede zu den Permutationen COLAMD und SFC sehr gering.

Somit wurde gezeigt, dass mit Hilfe effizienter numerischer Verfahren TsunAWI-NH trotz des erheblich höheren Rechenaufwands auch bei komplexen Tsunamiszenarien eingesetzt werden kann. Bei der Simulation des Tōhoku-Szenario auf einem globalen Gitter wird mit 64 Prozessoreinheiten in 369 min Gesamtlaufzeit eine Zeitintegration von 30 h berechnet. Zum Vergleich: Die OpenMP-parallele Produktivversion benötigt für das hydrostatische Flachwassermodell bei 32 Threads 347 min. Mit einer höheren

Anzahl von Prozessoreinheiten berechnet demnach TsunAWI-NH unter Berücksichtigung des nichthydrostatischen Druckanteils komplexe Tsunamisimulationen in einer annehmbaren Gesamtlaufzeit. Bei den im Rahmen dieser Arbeit berechneten Simulationen von verschiedenen realen Tsunamireignissen ist der Unterschied zwischen den Modellergebnissen von TsunAWI und TsunAWI-NH sehr gering. An den Stellen, für die Pegelaufzeichnungen als Referenzdaten verfügbar sind, ist die Welle noch weit davon entfernt zu brechen, so dass sie sich immer noch im Bereich der Flachwasserbewegung befindet und so auch vom Flachwassermodell dargestellt werden kann. Jedoch können einige Aussagen getroffen werden: Bei der Wellenausbreitung nimmt die maximale Wellenhöhe im nichthydrostatischen Modell früher ab. Im Vergleich mit Pegelständen wird die führende Tsunamiwelle von beiden Modellen gleich abgebildet. Dies deckt sich mit den Ankunftszeiten, die nur minimale Unterschiede aufweisen. Erst bei folgenden Wellenbergen, die Reflexionen und Überlagerungen in sich tragen, weichen die Modellergebnisse voneinander ab. Es findet häufig eine Phasenverschiebung statt, wobei sich die nichthydrostatisch modellierte Welle langsamer bewegt. Da zu diesem Zeitpunkt beide Modellergebnisse von den Messwerten abweichen, kann nicht festgestellt werden, welches der beiden Modelle in diesem Fall besser ist. Andere Faktoren, wie die Wahl der optimalen Anfangsbedingungen und Quellterme zeigen viel höhere Auswirkungen auf die Modellergebnisse.

Die im Rahmen dieser Arbeit durchgeführten Anwendungen umfassen idealisierte und Laborexperimente mit deutlich verbesserten Modellergebnissen sowie realistische Szenarien mit effizient eingesetzten numerischen Verfahren für Tsunamisimulationen bis hin zur globalen Skala. Um spezifische, komplexe Situationen, wie gerade das Mittelmeer mit seinen steilen Bathymetrieanstiegen beschreiben zu können, sind weitere Schritte zu unternehmen. Das Mittelmeer weist eine besondere Dichteschichtung auf. Dabei spielen starke Verdunstung und einströmendes salzärmeres Atlantikwasser eine entscheidende Rolle. Diese Verhältnisse wirken sich zusätzlich zur außergewöhnlichen Bathymetrie auf die Wellenausbreitung aus. Um dieser Situation gerecht zu werden kann zum Beispiel ein dreidimensionales hydrodynamisches Modell aufgesetzt werden, statt mit den vertikal gemittelten Größen zu rechnen. Dieser Ansatz ist jedoch sehr rechen- und speicherintensiv. Dennoch könnte ermittelt werden, ob die hier verwendeten numerischen Methoden auch für das Lösen von Gleichungssystemen dreidimensionaler Problemstellung geeignet sind. Ein anderer Ansatz bietet ein vertikal gemittelttes mehrschichtiges Modell, wie in [62] im FD Modell gezeigt. Hierbei wird für jede Schicht ein Gleichungssystem gelöst. Dies dürfte unter Einsetzen der hier untersuchten Methoden keine Herausforderung darstellen. Zudem könnte der Fundus der hier verwendeten Vorkonditionierungsmethoden um den Multiplicative Schwarz-Algorithmus erweitert werden. Während TsunAWI vom Ozeanmodell FEOM abgeleitet wurde, so kann nun rückführend das anhand von TsunAWI-NH erlangte Verständnis für effiziente Lösungsverfahren auf die großen, dünnbesetzten, verteilten Gleichungssysteme angewendet werden, die in FESOM (Finite Element Sea-ice Ocean Model) bei der Simulation von Ozean-Zirkulationen mit Meereis-Interaktion auftauchen.

A. Herleitung der verwendeten Gleichungen

A.1. Vertikal gemittelte nichthydrostatische Druckgradienten

Für den nichthydrostatischen Druckgradienten wird nun die Veränderung in x -Richtung betrachtet, die Änderung in y -Richtung kann analog berechnet werden. Mit Hilfe der Leibnizregel für Parameterintegrale kann

$$\overline{\partial_x p'} = \frac{1}{H} \int_{-h}^{\eta} \partial_x p' dz = \frac{1}{H} \partial_x (\overline{p'} H) - \frac{1}{H} p'(\eta) \partial_x \eta - \frac{1}{H} p'(-h) \partial_x h$$

geschrieben werden. Da p' an Oberfläche und Grund die Werte 0 und q annimmt, folgt durch Ersetzen des Integrals der rechten Seite durch (2.4) und das Anwenden der Produktregel,

$$\begin{aligned} \overline{\partial_x p'} &= \frac{1}{H} \partial_x \left(\frac{q}{2} H \right) - \frac{q}{H} \partial_x h, \\ &= \partial_x \frac{q}{2} + \frac{q}{2H} (\partial_x \eta - \partial_x h). \end{aligned} \tag{A.1}$$

Die Veränderung des Verhältnisses $\bar{q} := q/H$ in horizontaler Richtung entspricht nach der Quotientenregel

$$\partial_x \bar{q} = \partial_x \left(\frac{q}{H} \right) = \frac{H \partial_x q - q \partial_x H}{H^2} = \frac{\partial_x q}{H} - \frac{\bar{q}}{H} (\partial_x \eta + \partial_x h),$$

so dass die Veränderung des dynamischen Bodendrucks auch durch

$$\partial_x q = H \partial_x \bar{q} + \bar{q} (\partial_x \eta + \partial_x h)$$

dargestellt werden kann. Eingesetzt in Gleichung (A.1) führt dies zu

$$\overline{\partial_x p'} = \frac{H}{2} \partial_x \bar{q} + \frac{\bar{q}}{2} (\partial_x \eta + \partial_x h) + \underbrace{\frac{q}{2H}}_{=\bar{q}/2} (\partial_x \eta - \partial_x h) = \frac{H}{2} \partial_x \bar{q} + \bar{q} \partial_x \eta.$$

Für die gemittelte Druckänderung in vertikaler Richtung gilt durch die Auswertung von p' an den Integralgrenzen,

$$\overline{\partial_z p'} = \frac{1}{H} \int_{-h}^{\eta} \partial_z p' dz = \frac{1}{H} (p'(\eta) - p'(-h)) = -\frac{q}{H} = -\bar{q}.$$

A.2. Flachwassergleichungen

Über die Tiefe integriert, gilt für die Kontinuitätsgleichung

$$0 = \int_{-h}^{\eta} \partial_x v_x + \partial_y v_y + \partial_z v_z dz = \int_{-h}^{\eta} \partial_x v_x dz + \int_{-h}^{\eta} \partial_y v_y dz + \int_{-h}^{\eta} \partial_z v_z dz.$$

Mit Hilfe der Leibnizregel kann folgende Umformung getätigt werden:

$$\begin{aligned} 0 = & \partial_x \int_{-h}^{\eta} v_x dz - v_x(-h) \partial_x h - v_x(\eta) \partial_x \eta \\ & + \partial_y \int_{-h}^{\eta} v_y dz - v_y(-h) \partial_y h - v_y(\eta) \partial_y \eta \\ & + \underbrace{v_z(\eta) - v_z(-h)}_{w_\eta - w_{-h}}, \end{aligned}$$

wobei der auf die vertikale Geschwindigkeit bezogene Term an den Integralgrenzen η und $-h$ ausgewertet wird. Diese Werte sind von den kinematischen Randbedingungen

$$\begin{aligned} w_\eta &= \partial_t \eta + v_x(\eta) \partial_x \eta + v_y(\eta) \partial_y \eta \\ w_{-h} &= -\partial_t h - v_x(-h) \partial_x h - v_y(-h) \partial_y h \end{aligned}$$

reguliert, welche mit vertikal gemittelten Werten in den Gleichungen (2.8) und (2.9) dargestellt sind. Die Differenz der Werte an Oberfläche und Grund lautet demnach

$$w_\eta - w_{-h} = \partial_t \eta + v_x(\eta) \partial_x \eta + v_y(\eta) \partial_y \eta + \partial_t h + v_x(-h) \partial_x h + v_y(-h) \partial_y h,$$

und die Kontinuitätsgleichung kann mit

$$\partial_x \int_{-h}^{\eta} v_x dz + \partial_y \int_{-h}^{\eta} v_y dz + \partial_t (\eta + h) = 0$$

bestimmt werden. Mit der Verwendung des in (2.7) beschriebenen, gemittelten Geschwindigkeitsvektors \mathbf{u} und der Annahme $\partial_t h = 0$ ergibt dies,

$$\partial_t \eta + \nabla \cdot (\mathbf{u}H) = 0.$$

A.3. Korrekturterme

Die Bewegungsgleichung (2.17) kann umgeformt werden zu

$$\partial_t \mathbf{u} = \mathbf{F}_{xy} - (\mathbf{u} \cdot \nabla) \mathbf{u} - g \nabla \eta - \overline{\nabla p'}.$$

Mit der Lösung $\tilde{\mathbf{u}}$ der Gleichung (2.21) für $\bar{q} \equiv 0$, folgt

$$\partial_t \mathbf{u} = \partial_t \tilde{\mathbf{u}} - \overline{\nabla p'},$$

und mit Einsetzen von Gleichung (2.13) in diskretisierter Form

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^{n-1}}{2\Delta t} = \frac{\tilde{\mathbf{u}}^{n+1} - \mathbf{u}^{n-1}}{2\Delta t} - \left(\frac{H^n}{2} \nabla \bar{q}^{n+1} + \bar{q}^{n+1} \nabla \eta^n \right).$$

Folglich kann die horizontale Geschwindigkeit \mathbf{u} über das Zwischenergebnis $\tilde{\mathbf{u}}$ mit

$$\mathbf{u}^{n+1} = \tilde{\mathbf{u}}^{n+1} - \Delta t \left(H^n \nabla \bar{q}^{n+1} + 2\bar{q}^{n+1} \nabla \eta^n \right).$$

beschrieben werden. Analog kann mit einer Zwischenlösung \tilde{w} der Gleichung (2.22) für $q \equiv 0$ der Zusammenhang

$$\partial_t w = \partial_t \tilde{w} - \overline{\partial_z p'}$$

beschrieben werden. In diskretisierter Form und aufgelöst nach w^{n+1} , bedeutet das:

$$w^{n+1} = \tilde{w}^{n+1} + 2\Delta t \bar{q}^{n+1}.$$

A.4. Schwache Formulierung der Kontinuitätsgleichung

Durch Multiplikation mit einer Testfunktion $\varphi(x, y) \in V_h$ kann die Kontinuitätsgleichung (2.15) im Volumenintegral in die schwache Formulierung

$$\int_V (\partial_x v_x + \partial_y v_y + \partial_z v_z) \varphi \, dV = 0 \quad (\text{A.2})$$

überführt werden. Horizontale und vertikale Geschwindigkeitskomponenten werden wiederum getrennt betrachtet. Im Falle der horizontalen Geschwindigkeitskomponenten gilt für die Komponente in x -Richtung:

$$\begin{aligned} \int_V (\partial_x v_x) \varphi \, dV &= \int_{-h}^{\eta} \int_{\Omega} (\partial_x v_x) \varphi \, d\Omega dz, \\ &= - \int_{-h}^{\eta} \int_{\Omega} v_x (\partial_x \varphi) \, d\Omega dz + \int_{-h}^{\eta} \int_{\Gamma} v_x n_x \varphi \, d\Gamma dz, \\ &= - \int_{\Omega} \partial_x \varphi \int_{-h}^{\eta} v_x \, dz d\Omega + \int_{\Gamma} n_x \varphi \int_{-h}^{\eta} v_x \, dz d\Gamma, \\ &= - \int_{\Omega} (\partial_x \varphi) u H \, d\Omega + \int_{\Gamma} n_x \varphi u H \, d\Gamma. \end{aligned}$$

Hierbei beschreibt n_x die x -Komponente des vom Rand nach außen zeigenden Orthogonalenvektors. Bei dieser Umformung wird zuerst partiell integriert und anschließend mit dem Satz von Fubini die Reihenfolge der Integrale vertauscht. Daraufhin wird der über die Tiefe gemittelte Wert u aus (2.7) eingesetzt. Nach dem analogen Aufstellen dieser Gleichung in y -Richtung kann folgende Zusammenfassung getätigt werden,

$$\int_V (\partial_x v_x + \partial_y v_y) \varphi \, dV = - \int_{\Omega} \mathbf{u} \cdot \nabla \varphi H \, d\Omega + \int_{\Gamma_o} \mathbf{u} \cdot \mathbf{n} H \varphi \, d\Gamma_o + \int_{\Gamma_u} \mathbf{u} \cdot \mathbf{n} H \varphi \, d\Gamma_u, \quad (\text{A.3})$$

A. Herleitung der verwendeten Gleichungen

wobei das Randintegral für offene und undurchlässige Ränder separat betrachtet wird. Mit der Randbedingung (3.13) verschwindet das Integral entlang der undurchlässigen Ränder aus Γ_u . Für die Veränderung der vertikalen Geschwindigkeitskomponente gilt:

$$\begin{aligned} \int_V \partial_z v_z \varphi \, dV &= \int_{\Omega} \int_{-h}^{\eta} \partial_z v_z \varphi \, dz d\Omega, \\ &= \int_{\Omega} \varphi \int_{-h}^{\eta} \partial_z v_z \, dz d\Omega, \\ &= \int_{\Omega} \varphi (w_{\eta} - w_{-h}) \, d\Omega. \end{aligned} \tag{A.4}$$

Mit der Annahme (2.10) entspricht die Differenz $w_{\eta} - w_{-h} = 2(w - w_{-h})$. Mit den Gleichungen (A.3) und (A.4) kann (A.2) als

$$\begin{aligned} 0 &= \int_V (\partial_x v_x + \partial_y v_y + \partial_z v_z) \varphi \, dV \\ &= \int_{\Omega} -(\mathbf{u} \cdot \nabla \varphi) H + 2(w - w_{-h}) \varphi \, d\Omega + \int_{\Gamma_o} \mathbf{u} \cdot \mathbf{n} H \varphi \, d\Gamma_o \end{aligned}$$

dargestellt werden. Da die Randbehandlung mit (3.23) an offenen Rändern durch eine Dirichlet-Bedingung kontrolliert wird, wird die Gleichung an den entsprechenden Stellen nicht verwendet und das Randintegral kann im weiteren unbeachtet bleiben. Durch Einsetzen der Korrekturterme (3.17), (3.18) kann folgende Umformung getätigt werden:

$$\begin{aligned} 0 &= \int_{\Omega} -(\tilde{\mathbf{u}} - \Delta t (H \nabla \bar{q} + 2\bar{q} \nabla \eta)) \cdot \nabla \varphi H + 2(\tilde{w} + 2\Delta t \bar{q} - w_{-h}) \varphi \, d\Omega, \\ &= - \int_{\Omega} \tilde{\mathbf{u}} \cdot \nabla \varphi H \, d\Omega + \Delta t \int_{\Omega} H^2 \nabla \bar{q} \cdot \nabla \varphi + 2\bar{q} H \nabla \eta \cdot \nabla \varphi \, d\Omega \\ &\quad + \int_{\Omega} 2(\tilde{w} - w_{-h}) \varphi \, d\Omega + 4\Delta t \int_{\Omega} \bar{q} \varphi \, d\Omega. \end{aligned}$$

Nach einer Umsortierung der Terme und Multiplikation der Gleichung mit dem Faktor $1/\Delta t$ folgt:

$$\int_{\Omega} H^2 \nabla \bar{q} \cdot \nabla \varphi + 2\bar{q} H \nabla \eta \cdot \nabla \varphi + 4\bar{q} \varphi \, d\Omega = \frac{1}{\Delta t} \int_{\Omega} \tilde{\mathbf{u}} \cdot \nabla \varphi H - 2(\tilde{w} - w_{-h}) \varphi \, d\Omega.$$

A.5. Assemblierung

Gleichung (3.21) wird für die Ansatzfunktionen $\varphi_i \in \mathcal{F}_{\mathcal{N}}$ als Testfunktion aufgestellt und in die Darstellung

$$\sum_{j=1}^N a_{ij} \bar{q}_j = b_i, \quad 1 \leq i \leq N,$$

übergeführt. Da die Testfunktion $\varphi_i \in \mathcal{F}_{\mathcal{N}}$ einen kompakten Träger besitzt, kann das Integral über dem Gesamtgebiet Ω auf eine Summe von Teilintegralen über diejenigen

Elemente reduziert werden, die im Träger T_i von φ_i enthalten sind. Für die linke Seite der Gleichung (3.21) gilt demnach

$$\begin{aligned} & \int_{\Omega} H^2 \nabla \bar{q} \cdot \nabla \varphi_i + 2H\bar{q} \nabla \eta \cdot \nabla \varphi_i + 4\bar{q} \varphi_i d\Omega \\ &= \sum_{\Omega^e \subset T_i} \int_{\Omega^e} \underbrace{H^2 \nabla \bar{q} \cdot \nabla \varphi_i}_{(i)} + \underbrace{2H\bar{q} \nabla \eta \cdot \nabla \varphi_i}_{(ii)} + \underbrace{4\bar{q} \varphi_i}_{(iii)} d\Omega^e, \end{aligned} \quad (\text{A.5})$$

wobei die Terme (i)-(iii) getrennt betrachtet werden. Mittels der Darstellung der Größen als Linearkombination gilt für Term (i):

$$\int_{\Omega^e} H^2 \nabla \bar{q} \cdot \nabla \varphi_i d\Omega^e = \int_{\Omega^e} \left(\sum_{m=1}^N H_m \varphi_m \right) \left(\sum_{n=1}^N H_n \varphi_n \right) \sum_{j=1}^N \bar{q}_j \nabla \varphi_j \cdot \nabla \varphi_i d\Omega^e,$$

Beschränkt auf ein Element Ω^e sind die Gradienten der Ansatzfunktionen $\nabla \varphi_i|_{\Omega^e} =: \nabla \varphi_{i,e}$ konstant und können deshalb vor das Integral gezogen werden:

$$\int_{\Omega^e} H^2 \nabla \bar{q} \cdot \nabla \varphi_i d\Omega^e = \sum_{j=1}^N \bar{q}_j \nabla \varphi_{j,e} \cdot \nabla \varphi_{i,e} \underbrace{\sum_{m=1}^N \sum_{n=1}^N H_m H_n \int_{\Omega^e} \varphi_m \varphi_n d\Omega^e}_{=: \lambda_j}$$

Mit der strikten Diagonalisierung - beschrieben in Abschnitt 3.3.1 - gilt für die Integrale

$$\int_{\Omega^e} \varphi_m \varphi_n d\Omega^e = \begin{cases} \delta_{mn} \frac{|\Omega^e|}{3} & : \Omega^e \in T_m \cap T_n \cap T_i, \\ 0 & : \text{sonst}, \end{cases} \quad (\text{A.6})$$

wobei δ_{mn} das Kroneckerdelta beschreibt. Mit den Werten $H_k^e = H(n_k^e)$ der Gesamtwassertiefe an den Elementknoten folgt:

$$\lambda_j := \nabla \varphi_{j,e} \cdot \nabla \varphi_{i,e} \sum_{k=1}^3 (H_k^e)^2 \frac{|\Omega^e|}{3}.$$

Dabei verschwinden mit $\nabla \varphi_{j,e}$ alle Summanden λ_j für die $\Omega^e \not\subset T_i \cap T_j$ gilt. Mit Gleichung (A.6) folgt analog für die Terme (ii) und (iii):

$$\int_{\Omega^e} 2H\bar{q} \nabla \eta \cdot \nabla \varphi_i d\Omega^e = 2\nabla \eta_e \cdot \nabla \varphi_{i,e} \sum_{j=1}^N \sum_{k=1}^N \bar{q}_j H_k \int_{\Omega^e} \varphi_j \varphi_k d\Omega^e =: \sum_{j=1}^N \bar{q}_j \mu_j$$

mit

$$\mu_j = \begin{cases} 2\nabla \eta_e \cdot \nabla \varphi_{i,e} H_j \frac{|\Omega^e|}{3} & : \Omega^e \subset T_i \cap T_j, \\ 0 & : \text{sonst} \end{cases}$$

und

$$\int_{\Omega^e} 4\bar{q} \varphi_i d\Omega^e = 4 \sum_{j=1}^N \bar{q}_j \int_{\Omega^e} \varphi_j \varphi_i d\Omega^e = 4\bar{q}_i \frac{|\Omega^e|}{3}.$$

A. Herleitung der verwendeten Gleichungen

Mit den berechneten Koeffizienten kann Gleichung (A.5) mit

$$\begin{aligned}
& \int_{\Omega} H^2 \nabla \bar{q} \cdot \nabla \varphi_i + 2H \bar{q} \nabla \eta \cdot \nabla \varphi_i + 4\bar{q} \varphi_i d\Omega \\
&= \sum_{\Omega^e \subset T_i} \sum_{j=1}^N \left(\lambda_j + \mu_j + 4\delta_{ij} \frac{|\Omega^e|}{3} \right) \bar{q}_j \\
&= \sum_{j=1}^N \underbrace{\sum_{\Omega^e \subset T_i \cap T_j} \left(\lambda_j + \mu_j + 4\delta_{ij} \frac{|\Omega^e|}{3} \right)}_{=: a_{ij}} \bar{q}_j
\end{aligned}$$

dargestellt werden, da die Koeffizienten λ_j und μ_j für $\Omega^e \not\subset T_i \cap T_j$ verschwinden. Somit gilt für die Matrixeinträge:

$$a_{ij} = \sum_{\Omega^e \subset T_i \cap T_j} \left(\sum_{k=1}^3 (H_k^e)^2 \nabla \varphi_{j,e} \cdot \nabla \varphi_{i,e} + 2H_j \nabla \eta_e \cdot \nabla \varphi_{i,e} + 4\delta_{ij} \right) \frac{|\Omega^e|}{3}$$

Die rechte Seite der Gleichung (3.21) kann ebenfalls als eine Summe von Teilintegralen über Elemente des Trägers T_i dargestellt werden. Mit der Schreibweise der Größen als Linearkombination beschränkt auf das Element Ω^e und der Verwendung der Massenmatrizen $\mathbf{F}^e \in \mathbb{R}^{3 \times 3}$ mit den Einträgen

$$f_{kj}^e = (\delta_{kj} + \delta_{kl}) \frac{|\Omega^e|}{6}, \quad l = \text{mod}(j, 3) + 1$$

aus Tabelle 3.1 folgt

$$\begin{aligned}
b_i &= \frac{1}{\Delta t} \int_{\Omega} H \tilde{\mathbf{u}} \cdot \nabla \varphi_i - 2(\tilde{w} - w_{-h}) \varphi_i d\Omega \\
&= \frac{1}{\Delta t} \sum_{\Omega^e \subset T_i} \int_{\Omega^e} \sum_{j=1}^3 H_j^e \varphi_j^e \sum_{k=1}^3 \tilde{\mathbf{u}}_k^e \psi_k^e \cdot \nabla \varphi_i d\Omega^e - 2 \int_{\Omega^e} \sum_{m=1}^N (\tilde{w}_m - w_{-h,m}) \varphi_m \varphi_i d\Omega^e, \\
&= \frac{1}{\Delta t} \sum_{\Omega^e \subset T_i} \sum_{j=1}^3 \sum_{k=1}^3 H_j^e \tilde{\mathbf{u}}_k^e \underbrace{\int_{\Omega^e} \psi_k^e \varphi_j^e d\Omega^e}_{=f_{kj}^e} \cdot \nabla \varphi_{i,e} - 2 \sum_{m=1}^N (\tilde{w}_m - w_{-h,m}) \int_{\Omega^e} \varphi_m \varphi_i d\Omega^e \\
&= \frac{1}{\Delta t} \sum_{\Omega^e \subset T_i} \left(\sum_{j=1}^3 \sum_{k=1}^3 (\delta_{kj} + \delta_{kl}) H_j^e \tilde{\mathbf{u}}_k^e \cdot \nabla \varphi_{i,e} - 4(\tilde{w}_i - w_{-h,i}) \right) \frac{|\Omega^e|}{6},
\end{aligned}$$

wobei $H_j^e = H(n_j^e)$, $\tilde{\mathbf{u}}_k^e = \tilde{\mathbf{u}}(k_k^e)$ die Größen an den Eckknoten beziehungsweise Kantenmittelpunkten des Elements Ω^e wiedergeben.

Bezeichnungen

Abkürzungen und Akronyme

AWI	Alfred-Wegener-Institut, Helmholtz-Zentrum für Polar- und Meeresforschung
BJ	Block-Jacobi
COLAMD	COLumn Approximate Minimum Degree ordering
CRS	Compressed Row Storage
FD, FE, FV	Finite Differenzen, Finite Elemente, Finite Volumen
(F)GMRES	(Flexible) Generalized Minimum RESidual algorithm
GITEWS	German Indonesian Tsunami Early Warning System
HT	Householder-Transformationen
IOC	Intergovernmental Oceanographic Commission
JMA	Japan Meteorological Agency
LINZ	Land Information New Zealand
mGS	modifiziertes Gram-Schmidt-Verfahren
MPI	Message Passing Interface
NOAA	National Oceanic and Atmospheric Administration
OpenMP	Open Multi Processing
PE	Prozessoreinheit
RAS	Restricted Additive Schwarz
RCM	Reverse Cuthill-McKee
SFC	Space Filling Curve
SHOA	Servicio Hidrográfico y Oceanográfico de la Armada de Chile

Symbole

$\mathbf{A} \in \mathbb{R}^{N \times N}$, $\mathbf{A}_v \in \mathbb{R}^{N_v \times N_v}$	Matrix des Gesamt- bzw. reduzierten Teilsystems
A_h	horizontaler Viskositätskoeffizient in $[\text{m}^2/\text{s}]$
$\mathbf{b} = (b_i) \in \mathbb{R}^N$	rechte Seite
c, c_g	Phasen- und Gruppengeschwindigkeit in $[\text{m}/\text{s}]$
δ_{ij}	Kroneckerdelta
$\delta_e \in [-1, 1]$	empirischer Korrelationskoeffizient
$\partial_t, \partial_x, \partial_y, \partial_z$	Partielle Ableitung nach t, x, y, z
$\nabla = (\partial_x, \partial_y)^T$	2D-Gradient

Symbole

$E(x, x_{\text{ref}}) \in \mathbb{R}$	Effektivwert
η	Oberflächenauslenkung in [m]
$\eta_0, \mathbf{u}_0, w_0$	Anfangsbedingungen
η_{max}	maximale Wellenhöhe in [m]
F_x, F_y, F_z	spezifische Kraftkomponenten in [m/s ²]
f	Coriolisparameter in [1/s]
$f_{LU} \in \mathbb{R}$	Füllfaktor
g	Erdbeschleunigung in [m/s ²]
$\Gamma, \Gamma_u, \Gamma_o$	Rechengebietsränder
$H = \eta + h$	Gesamtwassertiefe in [m]
h	Topographie/Bathymetrie in [m]
\mathbf{K}^{-1}	Vorkonditionierungsoperator
$\kappa(t)$	Anteil des Teilgebiets Ω_v in Ω
$\mathcal{K}^i = \mathcal{K}^i(\mathbf{A}; \mathbf{x}_0) \subset \mathbb{R}^n$	Krylov-Unterraum
λ	Wellenlänge in [m]
$\mathbf{M}^e, \mathbf{F}^e, \mathbf{L}^e \in \mathbb{R}^{3 \times 3}$	Massenmatrizen bezüglich des Elements Ω^e
\mathcal{N}, \mathcal{K}	Stützstellenmengen
$\varnothing n_{\text{it}}, \varnothing t_{\text{it}}$	gemittelte Kenngrößen
$\mathbf{n}(\mathbf{x}), \mathbf{x} \in \Gamma$	nach außen gerichteter Orthonormalenvektor
$\mathcal{N}_v(t), \mathcal{N}_w(t), \mathcal{N}_k \subset \mathcal{N}$	Teilmengen der Gitterknotenmenge
$\Omega, \Omega_v \in \mathbb{R}^2$	Gesamtrechengebiet und variables Teilgebiet
$\Omega^e \in \mathcal{T}$	Element der Triangulierung \mathcal{T}
$p = p_0 + p'$	Druck pro Dichte in [m ² /s ²]
p_0, p', q	Druckanteile pro Dichte in [m ² /s ²]
Π_p	Gebietszerlegung in p Partitionen.
$\varphi_i \in \mathcal{F}_{\mathcal{N}}, \psi_j \in \mathcal{F}_{\mathcal{K}}$	Ansatzfunktionen
$\mathcal{P}^1, \mathcal{P}_{\text{NC}}^1$	Raum der (nicht-)konformen linearen Polynome
$\bar{q} = q/H$	Dynamischer Bodendruck pro Gesamtwassertiefe in [m/s ²]
$\mathbf{r}_i = \mathbf{b} - \mathbf{A}\mathbf{x}_i \in \mathcal{K}^{i+1}$	Residuum
ρ	Dichte in [kg/m ³]
S, S_A	Indexmengen
t	Zeit in [s]
t_a	Ankunftszeit der Welle in [s]
t_{end}	Integrationszeit in [s]
$t_{\text{konf}}, t_{\text{LGS}}, t_{\text{init}}, t_{\text{loop}}$	Rechenzeiten in [s]
$\mathcal{T} \subset \mathbb{R}^2$	zulässige Triangulierung
$\mathbf{u} = (u, v)$	gemittelte, horizontale Geschwindigkeit in [m/s]
v_x, v_y, v_z	Geschwindigkeitskomponenten in [m/s]
$\mathcal{V}_m = \{\mathbf{v}_1, \dots, \mathbf{v}_m\}$	Basis des Krylov-Unterraums \mathcal{K}^m
w, w_η, w_{-h}	vertikale Geschwindigkeitskomponenten in [m/s]
x, y, z	räumliche Koordinaten

Abbildungsverzeichnis

2.1	Chronologische Folge der Bewegungscharakteristiken eines Tsunamis.	5
2.2	Skizze der verwendeten Größen.	7
2.3	Skizze des (angenommenen) Verlaufs der Druckanteile.	8
2.4	Bahnkurven bei unterschiedlicher Wassertiefe.	9
2.5	Geschwindigkeitskomponenten bei unterschiedlicher Wassertiefe.	9
3.1	Das Standardreferenzelement Ω^R	16
3.2	Darstellung der Ansatzfunktionen.	17
3.3	Skizze der Approximationsfunktionen $\eta_{\mathcal{T}}$ und $\mathbf{u}_{\mathcal{T}}$	17
3.4	Approximation der Massenmatrix.	22
4.1	Modellaufbau zum Testfall einer stehenden Welle im Becken.	25
4.2	Vergleich von Ergebnissen hinsichtlich der Ausbreitungsgeschwindigkeit.	27
4.3	Zeitserie bei einer Beckentiefe von $h_b = 5$ m.	28
4.4	Modellaufbau für das Experiment mit linearem Anstieg zur Küste.	28
4.5	Vergleich von Modell- und Laborergebnissen im Beispiel $a_s = a_1$	30
4.6	Vergleich von Modell- und Laborergebnissen im Beispiel $a_s = a_2$	30
4.7	Empirischer Korrelationskoeffizient und Effektivwert für beide Beispiele.	31
4.8	Modellaufbau des Tankexperiments mit einem Unterwasserhindernis.	32
4.9	Vergleich von Modell- und Laborergebnissen.	34
4.10	Korrelationskoeffizient und Effektivwert in Bezug auf die Messwerte.	35
5.1	Überführung einer Triangulierung \mathcal{T} in den Graph $G_{\mathcal{T}}$	37
5.2	Skizze von Rechengebiet und Matrixstruktur.	38
5.3	Die Beschreibung der Matrix aus Abbildung 5.2 in CRS-Darstellung.	39
5.4	Veränderung der Matrixstruktur bei der Gauß-Elimination.	40
5.5	Veränderung des Graphen bei der Eliminierung eines Eintrags.	40
5.6	Sortierung mit Hilfe einer Hilbertkurve.	41
5.7	Charakteristische Füllmuster der verschiedenen Nummerierungen.	41
5.8	Das variable Rechengebiet Ω_v ist ein Teilgebiet des Gesamtgebiets Ω .	42
5.9	Die Belegungsstruktur der Matrix \mathbf{A}	43
5.10	Die Belegungsstrukturen der Matrix \mathbf{A}_v	44
6.1	Zerlegung der Knotenmenge \mathcal{N} und des Rechengebiets Ω	45
6.2	Berechnung und Kommunikation der Werte im Rechengebiet.	47
6.3	Überführung der Triangulierung \mathcal{T} in den Hypergraph $H_{\mathcal{T}}$	48
6.4	Skizze der globalen Umsortierung.	49
6.5	Zeilenweise verteiltes Gleichungssystem.	49

6.6	Klassifizierung der lokalen Knoten.	50
6.7	Zusammensetzung der lokalen Knotennummerierung	51
6.8	Charakteristische Strukturen bei der verteilten Matrix.	51
6.9	Belegungsstrukturen der verteilten Matrizen \mathbf{A} und \mathbf{A}_v	52
7.1	Schematische Darstellung der Minimierungsaufgabe von GMRES.	57
8.1	Iterative bei der Gauß-Elimination.	63
8.2	Matrixstrukturen bei der ILU(0)-Faktorisierung.	64
8.3	Blockbetrachtung der dünnbesetzten Matrix \mathbf{A}	66
8.4	Restriktion der Matrix \mathbf{A} auf den Diagonalblock \mathbf{A}_{22} . Die Blockmatrix I beschreibt die Einheitsmatrix.	66
8.5	Betrachtete Matrixeinträge beim BJ-Verfahren.	67
8.6	Überlappende Teilmatrizen bei der RAS-Präkonditionierung.	68
8.7	Beschränkung des Gleichungssystems auf die Grenzknoten.	70
8.8	Überführung des Gesamtsystems zum Schursystem.	70
8.9	Approximation des Schurkomplements.	71
9.1	Skizze der gemessenen Zeitspannen.	77
9.2	Die Größen η , n_{it} und t_{iter} bei verschiedenen ILU-Varianten.	80
9.3	Die Größen f_{LU} und $\varnothing t_{it}$ für verschiedene ILU-Faktorisierungen.	81
9.4	Gemessene Zeiten für Neuberechnete und wiederverwendete Strukturen.	82
9.5	Die Größen $\varnothing t_{konf}$, $\varnothing t_{it}$ und $\varnothing n_{it}$ bei verschiedener Vorkonditionierung.	84
9.6	Verteilung der Gitter- und Grenzkno bei verschiedenen Zerlegungen.	85
9.7	Die Größen t_{LGS} , s_p und e_p bei unterschiedlicher Vorkonditionierung.	86
9.8	Globale Knotennummerierung bei verschiedenen Sortierungen.	87
9.9	Die Größen $\varnothing t_{konf}$ und $\varnothing t_{it}$ je nach ILU/Sortierungs-Kombination.	88
9.10	Die Größen $\varnothing n_{it}$ und t_{loop} je nach ILU/Sortierungs-Kombination.	89
9.11	Lokale Knotennummerierung bei verschiedenen Permutationen.	89
9.12	Die Größen $\varnothing n_{it}$, t_{LGS} und t_{exp} bei unterschiedlicher Nummerierung.	90
9.13	Die Größen t_{loop} , s_p und e_p bei unterschiedlicher Gitternummerierung.	90
9.14	Epizentrumslokation und Initialauslenkung für ein Tsunamiszenario.	91
9.15	Die Größen $\varnothing t_{konf}$, $\varnothing t_{it}$, $\varnothing n_{it}$ und t_{loop} bei verschiedenen ILU-Varianten.	93
9.16	Die Größen $\varnothing t_{konf}$, $\varnothing t_{it}$ und $\varnothing n_{it}$ bei verschiedenen Vorkonditionierungen.	94
9.17	Die Größen t_{LGS} , s_p und e_p bei verschiedenen ILU-Faktorisierungen.	95
9.18	Die Zeiten t_{loop} , t_{LGS} und t_{exp} bei verschiedenen Gitternummerierungen.	95
9.19	Verhältnis $\kappa = \mathcal{N}_v / \mathcal{N} $ in Prozent.	96
9.20	Die Größen $\varnothing t_{konf}$, $\varnothing t_{it}$, $\varnothing t_{LGS}$ und f_{LU} bzgl. Gesamt- oder Teilsystem.	97
9.21	Die Zeiten $\varnothing t_{konf}$ und $\varnothing t_{it}$ bei der Berechnung des Teilsystems.	98
9.22	Verteilung der Knotenmenge \mathcal{N}_v^0 bei verschiedenen Zerlegungen.	98
9.23	Die Größen t_{LGS} , s_p und e_p für das Gesamt- oder Teilsystem.	99
9.24	Zerlegung der Gitterknoten bei unterschiedlicher Gewichtung.	100
9.25	Die Rechenzeiten t_{loop} , t_{LGS} und t_{exp} bei unterschiedlicher Gewichtung.	100
9.26	Verteilung der Knotenmengen \mathcal{N}_v^0 und \mathcal{N}	101
9.27	Die Zeiten $\varnothing t_{konf}$, t_{it} und t_{LGS} bei einfacher und zweistufiger Zerlegung.	103

9.28	Die Zeiten t_{LGS} und t_{exp} bei verschiedenen Zerlegungen.	103
9.29	Die Zeit t_{loop} und das Verhältnis t_{solve}/t_{loop} bei verschiedenen Vorkonditionierungen für Gesamt- und Teilsystem.	104
10.1	a) Die Position des Epizentrums und b) die Auslenkung η_0 [30].	107
10.2	Isolinien a) der maximalen Wellenhöhe η_{max} in Dezimeterschritten von 10-150 cm und b) der Ankunftszeit t_a im Stundentakt.	108
10.3	Zeitreihen mit gefilterten Daten der DART [®] -Bojen als Referenz.	110
10.4	Zeitreihen mit gefilterten Daten der Gezeitenpegel als Referenz.	110
10.5	Zeitreihen mit Pegelständen als Referenz, sowie die Distanz d_η	112
10.6	Isolinien bezüglich a) der Ankunftszeit t_a beziehungsweise b) der maximalen Wellenhöhe η_{max} im Abstand von 5 cm für 5-100 cm.	113
10.7	Isolinien beschreiben die Bathymetrie des Mittelmeeres.	114
10.8	Skizze der Bruchparameter.	115
10.9	Auslenkung η_0 für a) die Anregung O1 und b) die Anregung O2.	115
10.10	Isolinien der maximalen Wellenhöhe bei Anregung a) O1 und b) O2.	116
10.11	Die maximale Wellenhöhe bei den Anregungen a) O1 und b) O2.	117
10.12	Die Ankunftszeit $t_a(\mathbf{x}_p)$ bei den Anregungen a) O1 und b) O2.	117

Algorithmenverzeichnis

7.1	Iteratives Lösungsverfahren basierend auf der Richardson-Iteration.	53
7.2	Der modifizierte Gram-Schmidt-Algorithmus im Pseudo-Code.	55
7.3	Der GMRES-Algorithmus mit links- und rechtsseitiger Vorkonditionierung.	58
8.1	Die Gauß-Elimination zur vollständigen LU-Zerlegung und die Modifikationen zu den Varianten ILU(0), ILU(K) und ILUT im Pseudo-Code.	65
8.2	Schrittweise Vorwärts- und Rückwärtssubstitution, wobei Werte bezüglich innerer und Grenzknoten getrennt voneinander bestimmt werden.	72

Tabellenverzeichnis

2.1	Größenordnung charakteristischer Parameter einer Tsunamiwelle [36].	6
2.2	Ausbreitungsgeschwindigkeit c je nach Verhältnis H/λ	9
3.1	Auswertung der Integrale über Ω^e bezüglich der verschiedenen Kombinationen von Ansatzfunktionen.	22
4.1	Daten zum Modellaufbau der stehenden Welle im Becken.	26
4.2	Modellaufbau zum Tankexperiment mit linearem Anstieg zur Küste.	29
4.3	Modellaufbau des Tankexperiments mit einem Unterwasserhindernis.	33
4.4	Position der Messstationen des Tankexperiments.	34
9.1	Diskretisierungsparameter für das Testbeispiel der stehenden Welle.	79
9.2	Datenaustausch während Konfiguration und Anwendung.	84
9.3	Der Füllfaktor f_{LU} je nach Knotensortierung und ILU-Variante.	88
9.4	Diskretisierungsdaten zum Mentawai-Szenario.	92
9.5	Der Füllfaktor f_{LU} je nach ILU-Variante und Permutation.	92
9.6	Anteil der Grenzknoten innerhalb der Knotenmengen \mathcal{N} und \mathcal{N}_w^0	102
10.1	Diskretisierungsdaten des Tōhoku-Szenarios.	108
10.2	Daten zu Messstationen beim Tōhoku-Szenario.	109
10.3	Daten zu Messstationen beim Mentawai-Szenario.	112
10.4	Diskretisierungsdaten des Bourmedès-Zemmouri-Szenarios.	114
10.5	Bruchparameter zum Bourmedès-Zemmouri-Szenario: O1 [74], O2 [41]	115
10.6	Daten zu Messstationen beim Bourmedès-Zemmouri-Szenario.	116

Literaturverzeichnis

- [1] W. Albring. *Angewandte Strömungslehre*. Akademie Verlag GmbH, Berlin, 6. bearb. Auflage, 1990.
- [2] A. Androsov. Vergleich numerischer Methoden verschiedener Ordnung bei der Simulation eines Überströmungsexperiments. Persönliche Kommunikation, 2013.
- [3] A. Androsov, S. Harig, J. Behrens, J. Schröter, and S. Danilov. Tsunami modelling on unstructured grids: Verification and validation. In *Proceedings of the International Conference on Tsunami Warning (ICTW)*, Bali, Indonesia, 12-14 November 2008.
- [4] A. Androsov, S. Harig, A. Fuchs, A. Immerz, N. Rakowsky, W. Hiller, and S. Danilov. *Wave Propagation Theories and Applications*, chapter Tsunami wave propagation. INTECH, Croatia, 30 pp., 2013.
- [5] R. Asselin. Frequency filter for time integration. *Monthly Weather Review*, 100:487–490, 1972.
- [6] A. Y. Babeyko, A. Höchner, and S. V. Sobolev. Source modeling and inversion with near real-time GPS: a GITEWS perspective for Indonesia. *Natural Hazards and Earth System Sciences*, 10:1617–1627, 2010.
- [7] M. Bader. *Space filling Curves. An introduction with applications in scientific computing*. Springer-Verlag Berlin Heidelberg, 2013.
- [8] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. M. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. V. der Vorst. *Templates for the solution of linear systems: Building blocks for Iterative Methods, 2nd Edition*. SIAM, Philadelphia, 1993.
- [9] J. Behrens, A. Androsov, A. Y. Babeyko, S. Harig, F. Klaschka, and L. Mentrup. A new multi-sensor approach to simulation assisted tsunami early warning. *Natural Hazards and Earth System Sciences*, 10:1085–1100, 2010.
- [10] S. Beji and J. A. Battjes. Experimental investigation of wave propagation over a bar. *Coastal Engineering*, 19:151–162, 1993.
- [11] X.-C. Cai and M. Sarkis. A restricted additive Schwarz preconditioner for general sparse linear systems. *SIAM Journal on Scientific Computing*, 21:792–797, 1999.
- [12] V. Casulli. A semi-implicit finite difference method for non-hydrostatic, free-surface flows. *International Journal for Numerical Methods in Fluids*, 30:425–440, 1999.
- [13] Ü. V. Çatalyürek and C. Aykanat. Hypergraph-partitioning based decomposition

- for parallel sparse-matrix vector multiplication. *IEEE Trans. Parallel Distrib. Systems*, 10(7):673–693, 1999.
- [14] R. Courant, K. Friedrichs, and H. Lewy. Über die partiellen Differenzgleichungen der mathematischen Physik. *Mathematische Annalen*, 100:32–74, 1928.
- [15] H. Cui, J. D. Pietrzak, and G. S. Stelling. Improved efficiency of a non-hydrostatic, unstructured grid, finite volume model. *Ocean Modelling*, 54-55:55–67, 2012.
- [16] S. Danilov, G. Kivman, and J. Schröter. A finite-element ocean model: principles and evaluation. *Ocean Modelling*, 6(2):125–150, 2004.
- [17] T. A. Davis, J. R. Gilbert, S. I. Larimore, and E. G. Ng. A column approximate minimum degree algorithm. Technical Report TR-00-005, Department of Computer and Information Science and Engineering, University of Florida, 2000.
- [18] M. W. Dingemans. Comparison of computations with Boussinesq-like models and laboratory measurements. Technical Report H1684.12, Delft Hydraulics, 1994.
- [19] J. J. Dongarra, I. S. Duff, D. C. Sorensen, and H. A. van der Vorst. *Numerical Linear Algebra for High-Performance Computers*. Society for Industrial and Applied Mathematics, Philadelphia, 1998.
- [20] R. Fletcher. Conjugate gradient methods for indefinite systems. In G. A. Watson, editor, *Proceedings of the Dundee Biennial Conference on Numerical Analysis 1975*, pages 73–89. Springer, New York, 1976.
- [21] W. H. Frey and D. A. Field. Mesh relaxation: a new technique for improving triangulations. *International Journal for Numerical Methods in Engineering*, 31:1121–1133, 1991.
- [22] A. S. Furumoto, C. Morioka, and H. Tatehata. Japanese tsunami warning system. *Science of Tsunami Hazards*, 17:85–105, 1999.
- [23] A. George and J. W. Liu. *Computer Solutions of Large Sparse Positive Definite Systems*. Prentice-Hall, Englewood Cliffs, New Jersey, 1981.
- [24] J. R. Gilbert, C. Moler, and R. Schreiber. Sparse matrices in MATLAB: Design and implementation. *SIAM Journal on Matrix Analysis*, 13(1):333–356, 1992.
- [25] G. H. Golub and C. F. van Loan. *Matrix computations*. The Johns Hopkins University Press, Baltimore, 1989.
- [26] J. Griffin, H. Latief, W. Kongko, S. Harig, N. Horspool, R. Hanung, A. Rojali, N. Maher, L. Fountain, A. Fuchs, J. Hossen, Supryati, S. E. Dewanto, and P. Cummins. An evaluation of onshore digital elevation models for tsunami inundation modelling. NH21C-1598. AGU Fall Meeting, San Francisco, California, 3-7 December 2012.
- [27] E. Hanert, D. Y. Le Roux, V. Legat, and E. Deleersnijder. An efficient Eulerian finite element method for the shallow water equations. *Ocean Modelling*, 10:115–136, 2005.

- [28] S. Harig, Chaeroni, W. S. Pranowo, and J. Behrens. Tsunami simulations on several scales. *Ocean Dynamics*, 58:429–440, 2008.
- [29] Helmholtz-Zentrum Potsdam Deutsches GeoForschungsZentrum. GEOFON. Homepage: <http://geofon.gfz-potsdam.de/eqinfo/>.
- [30] A. Höchner, A. Y. Babeyko, and S. V. Sobolev. Geodetic source model for the 2011 Tohoku earthquake and tsunami. In *Geophysical Research Abstracts*, 13, EGU2011-14272. General Assembly European Geosciences Union, Vienna, Austria, 3-8 April 2011.
- [31] F. Imamura, A. C. Yalciner, and G. Ozyurt. Tsunami modelling manual (TUNAMI model). April, 2006.
- [32] Intergovernmental Oceanographic Commission (IOC) of UNESCO. Sea Level Monitoring Facility. Homepage: <http://www.ioc-sealevelmonitoring.org/list.php>.
- [33] L. H. Kantha and C. A. Clayson. *Numerical Models of Oceans and Oceanic Processes*, volume 66 of *International Geophysics Series*. Academic Press, San Diego, 2000.
- [34] G. Karypis and V. Kumar. A fast and highly quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1999.
- [35] D. Y. Le Roux and C. A. Lin. A semi-implicit semi-lagrangian finite-element shallow-water ocean model. *Monthly Weather Review*, 128:1384–1401, 2000.
- [36] B. Levin and M. Nosov. *Physics of tsunamis*. Springer Science + Business Media B.V., 2009.
- [37] Z. Li and Y. Saad. SchurRAS: A restricted version of the overlapping Schur complement preconditioner. Report umsi-2004-76, University of Minnesota, Minnesota Supercomputer Institute, 2004.
- [38] Z. Li, Y. Saad, and M. Sosonkina. pARMS: a parallel version of the algebraic recursive multilevel solver. *Numerical Linear Algebra with Applications*, 10:485–509, 2003.
- [39] P. J. Lynett, T.-R. Wu, and P. L.-F. Liu. Modeling wave runup with depth-integrated equations. *Coastal Engineering*, 46:89–107, 2002.
- [40] G. R. Markall, A. Slemmer, D. A. Ham, P. H. J. Kelly, C. D. Cantwell, and S. J. Sherwin. Finite element assembly strategies on multi-core and many-core architectures. *International Journal for Numerical Methods in Fluids*, 71:80–97, 2013.
- [41] M. Meghraoui, S. Maouche, B. Chemaa, Z. Cakir, A. Aoudia, A. Harbi, P.-J. Alasset, A. Ayadi, Y. Bouhadad, and F. Benhamouda. Coastal uplift and thrust faulting associated with the $m_w = 6.8$ Zemmouri (Algeria) earthquake of 21 May, 2003. *Geophysical Research Letters*, 31(L19605), 2004.
- [42] A. Meister. *Numerik linearer Gleichungssysteme*. Friedr. Vieweg & Sohn Verlag, Wiesbaden, 2008.

- [43] U. Münch, A. Rudloff, and J. Lauterjung. Postface: The GITEWS project - results, summary and outlook. *Natural Hazards and Earth System Sciences*, 11:765–769, 2011.
- [44] National Oceanic and Atmospheric Administration’s National Data Buoy Center. Homepage: <http://www.ndbc.noaa.gov/dart.shtml>.
- [45] O. Nielsen, S. Roberts, D. Gray, A. McPherson, and A. Hitchman. Hydrodynamic modelling of coastal inundation. In *Proceedings of MODSIM05*, International Congress on Modelling and Simulation, Melbourne, Australia, 12-15 December 2003.
- [46] Y. Okada. Surface deformation due to shear and tensile faults in a half-space. *Bulletin of the Seismological Society of America*, 75(4):1135–1154, 85.
- [47] J. Pedlosky. *Geophysical Fluid Dynamics*. Springer Verlag, New York, 1987.
- [48] F. Pellegrini and J. Roman. SCOTCH: A software package for static mapping by dual recursive bipartitioning of process and architecture graphs. In *Proceedings of HPCN’96*, pages 493–498, 1996.
- [49] N. Rakowsky and A. Fuchs. Efficient local resorting techniques with space filling curves applied to the tsunami simulation model TsunAWI. In *GS 11, Abstracts*, MS 43, SIAM Conference on Mathematical and Computational Issues in the Geosciences, Long Beach, California, 21-24 März 2011.
- [50] N. Rakowsky, S. Harig, A. Immerz, A. Androsov, A. Fuchs, S. Danilov, W. Hiller, and J. Schröter. Operational tsunami modelling with TsunAWI - recent developments and applications. *Natural Hazards and Earth System Sciences*, 13:1629–1642, 2012.
- [51] A. J. Robert. The integration of a low order spectral form of the primitive meteorological equations. *Journal of the Meteorological Society of Japan*, 44(5), 1966.
- [52] A. Rudloff, J. Lauterjung, U. Münch, and S. Tinti. Preface: The GITEWS project (German-Indonesian Tsunami Early Warning System). *Natural Hazards and Earth System Sciences*, 9:1381–1382, 2009.
- [53] Y. Saad. A flexible inner-outer preconditioned GMRES algorithm. *SIAM J. Scientific Computing*, 14:461–469, 1993.
- [54] Y. Saad. *Iterative methods for sparse linear systems, Second Edition*. Society of Industrial and Applied Mathematics, Philadelphia, 2000.
- [55] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7:856–869, 1986.
- [56] Y. Saad and M. Sosenkina. Distributed Schur complement techniques for general sparse linear systems. *SIAM J. Sci. Comput.*, 21(4):1337–1356, 1999.
- [57] Y. Saad and H. A. van der Vorst. Iterative solution of linear systems in the 20th century. *Journal of Computational and Applied Mathematics*, 123:1–33, 2000.

- [58] A. Sahal, J. Roger, S. Allgeyer, B. Lemaire, H. Hébert, F. Schindelé, and F. Lavigne. The tsunami triggered by the 21 May 2003 Boumerdès-Zemmouri (Algeria) earthquake: field investigations on the French Mediterranean coast and tsunami modelling. *Natural Hazards and Earth System Sciences*, 9:1823–1834, 2009.
- [59] T. Schöne, W. W. Pandoe, I. Mudita, S. Roemer, J. Illigner, C. Zech, and R. Galas. GPS water level measurements for Indonesia’s Tsunami Early Warning System. *Natural Hazards and Earth System Sciences*, 11:741–749, 2011.
- [60] J. R. Shewchuk. Triangle: Engineering a 2d quality mesh generator and delaunay triangulator. *Lecture Notes in Computer Science*, 1148:203–222, 1996.
- [61] R. B. Sidje. Alternatives for parallel Krylov subspace basis computations. *Numerical Linear Algebra with Applications*, 4(4):305–331, 1997.
- [62] G. S. Stelling and M. Zijlema. An accurate and efficient finite-difference algorithm for non-hydrostatic free-surface flow with application to wave propagation. *International Journal for Numerical Methods in Fluids*, 43:1–23, 2003.
- [63] C. E. Synolakis. *The runup of long waves*. PhD thesis, California Institute of Technology, Pasadena, California, 1986.
- [64] C. E. Synolakis. The runup of solitary waves. *J. Fluid Mech.*, pages 525–545, 1987.
- [65] C. E. Synolakis, E. N. Bernard, V. V. Titov, U. Kânoglu, and F. I. Gonzáles. Validation and verification of tsunami numerical models. *Pure and Applied Geophysics*, 165:2197–2228, 2008.
- [66] V. V. Titov and F. I. Gonzalez. Implementation and testing of the Method Of Splitting Tsunami (MOST) model. NOAA Technical Memorandum ERL PMEL-112, Pacific Marine Environmental Laboratory, 1997.
- [67] V. V. Titov, F. I. Gonzáles, H. O. Mofjeld, and J. C. Newman. Project SIFT (Short-term Inundation Forecasting for Tsunamis). In *ITS 2001 Proceedings, 7-2*, International Tsunami Symposium, Seattle, Washington, 7-10 August 2001.
- [68] V. V. Titov and C. E. Synolakis. Modeling of breaking and nonbreaking long-wave evolution and runup using VTCS-2. *Journal of Waterway, Port, Coastal and Ocean Engineering, ASCE*, 121(6):308–316, 1995.
- [69] United States Geological Survey. Homepage: <http://earthquake.usgs.gov>.
- [70] H. A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 13(2):631–644, 1992.
- [71] H. F. Walker. Implementation of the GMRES method using householder transformations. *SIAM J. Sci. Stat. Comput.*, 9(1):152–163, 1988.
- [72] R. A. Walters. A semi-implicit finite element model for non-hydrostatic (dispersive) surface waves. *International Journal for Numerical Methods in Fluids*, 49:721–737, 2005.

- [73] Y. Yamazaki, Z. Kowalik, and K. F. Cheung. Depth-integrated, non-hydrostatic model for wave breaking and run-up. *International Journal for Numerical Methods in Fluids*, 61:473–497, 2008.
- [74] A. Yelles, K. Lammali, A. Mahsas, E. Calais, and P. Briole. Coseismic deformation of the May 21st, 2003, $M_w = 6.8$ Bourmedes earthquake, Algeria, from GPS measurements. *Geophysical Research Letters*, 31(L13610), 2004.

Danksagung

Die Studien für diese Arbeit sowie das Schreiben der vorliegenden Dissertation wurden am Alfred-Wegener-Institut, Helmholtz-Zentrum für Polar- und Meeresforschung im Bereich Wissenschaftliches Rechnen durchgeführt. Vielen Dank meinem Betreuer Prof. Dr. Wolfgang Hiller für das jahrelange Begleiten und Unterstützen meiner Arbeit und dass die Tür immer für mich offen stand. Mein Dank gilt ebenso Prof. Dr. Alfred Schmidt für die Begutachtung dieser Dissertation und die hilfreichen Denkanstöße. Prof. Dr. Stephan Frickenhaus und Dr. Natalja Rakowsky danke ich, dass sie als Doktorandenkomitee bei unregelmäßigen Jour Fixe-Besprechungen den jeweils aktuellen Stand meiner Arbeit durch konstruktive Kritik beeinflussten und auch zwischen den Treffen immer Antworten auf meine Fragen fanden. Besonderen Dank verdient Dr. Sven Harig, der mir stets Zeit schenkte, um über meine Fortschritte, Fehlschläge und Ergebnisse zu diskutieren. Danke für die beständige Unterstützung und das Lesen und Kommentieren des gesamten Werkes. Dr. Alexey Androsov, Dr. Sergey Danilov und Dr. Jens Schröter halfen mir, die physikalische Sichtweise der Tsunamimodellierung nicht zu vernachlässigen und durch ihre Erklärungen besser zu verstehen, vielen Dank dafür. Danke ebenso an Dr. Haiyang Cui (TU Delft) für den fachlichen Austausch in puncto nichthydrostatische Modellierung. Vielen Dank auch an Dr. Martin Losch, der mir half über den Tellerrand hinauszublicken. Danke, Dr. Dirk Barbi, Dr. Lars Nerger und Dr. Bernadette Fritsch für die fachliche Kompetenz im Wissenschaftlichen Rechnen. Mein Dank geht auch an Prof. Dr. Guus Stelling und Dr. Marcel Zijlema von der TU Delft, da sie freundlicherweise die Messdaten zum Tankexperiment mit Unterwasserhindernis bereitstellten. Ich danke Prof. Dr. Jörn Behrens (KlimaCampus), Prof. Dr. Michael Bader (TU München), Prof. Dr. Francis X. Giraldo (Naval Postgraduate School) und Prof. Dr. Randall J. LeVeque (University of Washington) dafür, dass ich die Sommerschule in Monterey begleiten durfte, wo ich noch einiges über Tsunamimodellierung lernen konnte. Vielen Dank an Jörg Matthes, der sich meinen technischen Probleme stets angenommen hat. Danke, Claudia Wekerle und Dr.-Ing. Widodo Pranowo sowie Paul Kirchgessner für das angenehme Arbeitsklima. Antonia Immerz danke ich für die schöne Zeit, die wir auch außerhalb des Büros hatten. Zu guter Letzt bedanke ich mich bei Timo Kühnle, der mir stets den Rücken gestärkt und trotz fachlicher Abweichung jede Seite dieser Arbeit gelesen hat.