Datum: 1. Dezember 2025

Technische Universität Berlin

Institut für Mechanik

Numerische Implementierung der nichtlinearen FEM

Hausaufgabe 2

Modulverantwortliche: Prof. Dr.-Ing. Sandra Klinge
Betreuer: Stefan Hildebrandt

Gruppenname: Diskretisierer

Gruppenmitglieder:
Jinwook Choi 451789
Hannes König 512485
Valentin Lerch 402025
Sebastian Redemann 456703
WiSe 2025/26

Inhaltsverzeichnis

1 Methodik

1.1 Modalanalyse

Die Modalanalyse beschreibt das freie Schwingverhalten von Strukturen ohne eine Einwirkung durch äußere Kräfte. Mittels der Modalanalyse lassen sich die Eigenfrequenzen und die dazugehörigen Eigenmoden / Eigenschwingungsformen bestimmen und daraus Informationen über das Resonanzverhalten ableiten. Es sei an dieser Stelle angemerkt, dass die Schingung in einer Mode unabhängig beziehungsweise entkoppelt ist von den Schwingungen der restlichen Moden. Hintergrund ist die Orthogonalität der Eigenvektoren zueinander sowie der Eigenmoden bezüglich der Steifigkeits- und Massenmatrix. Eine freie Schwingung bedeutet, dass keine Dämpfung als auch äußere Belastung vorliegt. Die Bewegungsdifferentialgleichung lässt sich somit reduzieren zu

$$M \cdot \ddot{u} + K \cdot u = 0. \tag{1.1}$$

Mittels eines harmonischen Lösungansatzes, ergibt sich das allgemeine Matrizeneigenwertproblem

$$(K - \omega^2 M)u = 0. \tag{1.2}$$

Dieses lässt sich mittels zweier numerischer Lösungsansätze, entweder durch Inversion der Massenmatrix oder der Steifigkeitsmatrix, lösen. Es ergeben sich für den Ansatz der Inversion der Massenmatrix die Gleichung

$$I - \omega^2 K^{-1} M\varphi = 0 \tag{1.3}$$

und

$$A\varphi = \lambda\varphi \tag{1.4}$$

mit

$$A = M^{-}1K, \quad \lambda = \omega^2 \tag{1.5}$$

Für den alternativen Ansatz der Inversion der Steifigkeitsmatrix ergibt sich entsprechend

$$I - \omega^2 K^{-1} M\varphi = 0 \tag{1.6}$$

Durch Umformung erhält man die klassische Eigenwertdarstellung

$$K^{-1} M\varphi = \frac{1}{\omega^2}\varphi, \tag{1.7}$$

welche sich in kompakter Form schreiben lässt als

$$A\varphi = \lambda\varphi, \tag{1.8}$$

mit

$$A = K^{-1} M, \qquad \lambda = \frac{1}{\omega^2}. \tag{1.9}$$

Hierbei stellt λ_i den reziproken Eigenwert dar, aus dem die Eigenkreisfrequenzen nach

$$\omega_i = \frac{1}{\sqrt{\lambda_i}}$$

bestimmt werden. Dieses Vorgehen liefert die gleichen Eigenmoden wie die Inversion der Massenmatrix, unterscheidet sich jedoch in der numerischen Stabilität und der Skalierung der Eigenwerte.

1.2 Quadratische Elemente Q4 vs Q8/Q9

Für die erste Hausaufgabe wurde ein lineares quadratisches Element genutzt. Dieses Element ist auch unter dem Namen: Q4 (Abb. 1) bekannt. Der erste Knoten links unten wird hier mit 1 beziffert, die Zählkonvention läuft dann gegen den Uhrzeigersinn. Es besitzt nur Eckknoten und kann somit nur lineare Ansätze nutzen, was die Genauigkeit aber dafür auch den Rechenaufwand gering hält. [1, S. 126 ff.]

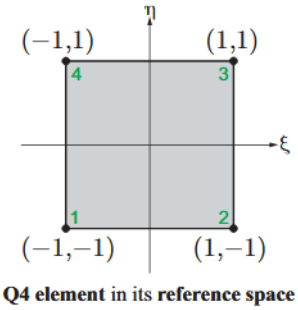


Abb. 1: Q4 Quadratisches Element [1]

Die in Abb. 2 gezeigten Q8- und Q9-Elemente erweitern dieses Konzept um Mittelknoten zwischen den Eckknoten bzw. einen zusätzlichen Zentrumsknoten. Dadurch entstehen quadratische Ansatzfunktionen, die gekrümmte Geometrien und Spannungsverläufe deutlich genauer abbilden können. Die Knotenfolge bleibt dabei konsistent im Umlaufsinn – bei Q8 über die Randknoten, bei Q9 mit zusätzlichem Mittelpunkt – was eine eindeutige Interpolation sicherstellt. Elemente mit Q8-Charakter, auch als Serendipity-Elemente bekannt, stehen den Q9-Elementen in der Genauigkeit kaum nach, sind dafür aber erkennbar weniger Rechenaufwendig. [1, S. 131 f.]

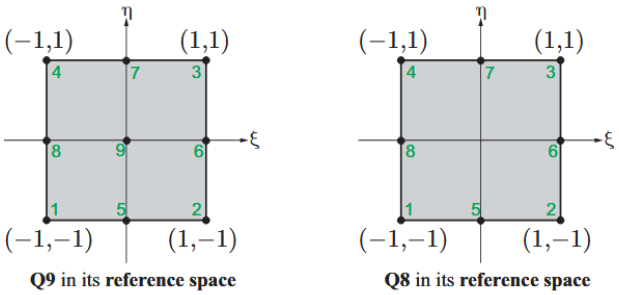


Abb. 2: Q8 und Q9 Quadratische Elemente [1]

1.2.1 Ansatzfunktionen

Die Ansatzfunktionen der Q4-Elemente basieren auf einer bilinearen Interpolation mit vier Eckknoten. Dies führt zu linearen Polynomen in ξ und η. Die Shape-Funktionen lauten:

$$N_i(\xi, \eta) = \frac{1}{4}(1 \pm \xi)(1 \pm \eta).$$

Dies ermöglicht eine einfache, aber grobe Approximation von Verschiebungen und Spannungen innerhalb des Elements, die nur konstante Gradienten abbildet. Im Gegensatz dazu nutzen Q8-Elemente (Serendipity) quadratische Ansatzfunktionen mit acht Knoten (vier Ecken und vier Mittelkanten). Die Shape-Funktionen erweitern sich zu:

$$N_i(\xi, \eta) = -\frac{1}{4}(1 \pm \xi)(1 \pm \eta)(1 \pm \xi\eta)$$

für Eckknoten und

$$N_i(\xi, \eta) = \frac{1}{2}(1 - \xi^2)(1 \pm \eta)$$

für Mittelkanten. Dadurch können gekrümmte Ränder und quadratische Verläufe präziser dargestellt werden, was die Genauigkeit bei komplexen Geometrien und Spannungssingularitäten verbessert. Der Mehraufwand ergibt sich aus der Verdopplung der Freiheitsgrade pro Element (von 8 auf 16 in 2D), was die Elementsteifigkeitsmatrix von 4×4 auf 8×8 (oder größer) aufbläht und die Integrationspunkte (z.,B. 3×3 statt 2×2) erhöht. Dies steigert den Rechenaufwand um etwa das Vierfache pro Element, bei vergleichbarer Genauigkeit zu Q9-Elementen mit Zentrumsknoten. [1, S. 126–132]

1.3 transiente Analyse

Die transiente Analyse untersucht das zeitabhängige Verhalten eines Systems, d.h. wie sich Verschiebungen, Spannungen, Beschleunigungen etc. über die Zeit verändern, wenn das System einer bestimmten Belastung (z.B. Kräften,

Stößen, Impulsen) ausgesetzt wird. Strukturantwort wird dargstellt in Abhängigkeit der Zeit. Um eine transiente Strukturantwort aufgrund äußerer zeitabhängiger Kräfte zu bestimmen, brauchen wir

$$M\ddot{u}_i + Cu_i + Ku_i = f_i(t) \tag{1.10}$$

über die Zeit integrieren. Dafür gibt es grundsätzlich 2 Optionen:

1. **Modalreduktion**: Gleichungssystem wird reduziert beziehungsweise in einen Unterraum transformiert mittels Transformationsmatrix. Effizienteste Transformationsmatrix: Modalmatrix (Spalten sind Eigenmoden; Moden sind orhtognonal zueinander bezüglich Massen- und Steifigkeitsmatrix). Somit ist eine Modalanalyse vorher Pflicht!
2. **Direkte Zeitintegration**: es findet keine Transformation in Unterraum statt. Numerisch sehr anspruchsvoll, kann jedoch durch eine vorherige Modalanalyse beeinflusst werden, um den Rechenaufwand vermindern. Eine Modalanalyse ist als erster Schritt jedoch grundsätzlich bei einer dynamischen Analyse eines Systems zu empfehlen.

Das Newmark-Beta-Verfahren gehört zur Kategorie der direkten Zeitintegration. Direkte Zeitintegration bedeutet, dass die Lösung ohne vorige Transformation in einen Unterraum, beziehungsweise ohne Modellreduktion, erfolgt. Wir nutzen also die sogenannten physikalischen Freiheitsgrade / Knotenverschiebungen. Die direkte Zeitintegration lässt sich unterscheiden in die **implizite** und **explizite** Zeitintegration. Das Newmark-Beta-Verfahren gehört zu der impliziten Zeitintegration.

1.3.1 Newmark-Beta-Verfahren

Das Newmark-Beta-Verfahren ist ein weit verbreitetes implizites Zeitintegrationsverfahren zur numerischen Lösung der Bewegungsgleichung elastischer Strukturen. Es basiert auf der Annahme, dass Verschiebung, Geschwindigkeit und Beschleunigung innerhalb eines Zeitschritts Δt durch eine vorgegebene Interpolationsform beschrieben werden, welche durch die beiden Parameter β und γ festgelegt wird. Für die häufig verwendete Wahl

$$\beta = \frac{1}{4}, \qquad \gamma = \frac{1}{2}, \tag{1.11}$$

ergibt sich das sogenannte Modell konstanter mittlerer Beschleunigung, das als unbedingt stabil gilt und daher in der strukturdynamischen FEM als Standardverfahren eingesetzt wird.

Die Grundidee des Verfahrens besteht darin, die Bewegungsgleichung nicht zu jedem Zeitpunkt exakt zu erfüllen, sondern lediglich in diskreten Zeitpunkten. Innerhalb eines Zeitschritts werden die kinematischen Größen mithilfe der Newmark-Integrationsformeln aktualisiert. Dadurch reduziert sich die Berechnung pro Zeitschritt auf die Lösung eines modifizierten quasistatischen Gleichgewichts unter Berücksichtigung von Trägheits- und Dämpfungskräften, sodass etablierte lineare statische Lösungsstrategien direkt angewendet werden können.

Typische Anwendungsschritte des Verfahrens umfassen:

- die Wahl der Zeitschrittweite Δt, abhängig von der maximal interessierenden Frequenz,
- die Festlegung der Gesamtdauer der Analyse und daraus resultierend der Anzahl der Zeitschritte,
- die iterative Aktualisierung von Verschiebungen, Geschwindigkeiten und Beschleunigungen durch Lösen eines modifizierten Gleichungssystems pro Zeitschritt.

Aufgrund seiner Stabilität, Flexibilität und vergleichsweise einfachen Implementierung zählt das Newmark-Beta-Verfahren zu den am häufigsten eingesetzten Verfahren der transienten FEM-Analyse.

2 Implementierung im Code

Um eine Stabilere Simulation zu erhalten wurde entgegengesetzt der ersten Abgabe, die Penalty-Methode von einem Steifigkeitswert von $10 \cdot 10^{22}$ auf einen moderateren, für den Speicher besser handhabbaren Wert von $10 \cdot 10^{13}$ gesetzt.

2.1 Parameterwahl für Balken und Netz

Quadratische Elementform aus Konvergenzgründen Um schnelle Anpassungen des Codes zu gewährleisten wurde ein ein Meter Balken, mit einem quadratischen Querschnitt von 100 mm pro Seite verwendet. Es wurde sich außerdem für ein gröberes Netz von 20 Elementen in x-Richtung und 2 Elementen in y-Richtung entschieden.

2.2 Modalanalyse

In diesem Abschnitt wird ausschließlich die numerische Implementierung der Modalanalyse beschrieben. Die physikalische Interpretation der Eigenfrequenzen und Eigenformen erfolgt in Kapitel 3.

Für die Dichte ρ wurde ein Wert von 7850 $\frac{\text{kg}}{\text{m}^3}$ gewählt. [2, S. 133]

2.2.1 Reduktion auf freie Freiheitsgrade

Die Einspannung an der linken Balkenseite wird im Code mittels einer Binärmaske `drlt_mask` verwaltet. Aus dieser Maske werden die Indizes der freien Freiheitsgrade `free_dofs` extrahiert. Um das Eigenwertproblem zu lösen, werden die globalen Systemmatrizen $\mathbf{K}$ und $\mathbf{M}$ mittels Tensor-Slicing auf die freien Freiheitsgrade reduziert:

$$\mathbf{K}_{\text{free}} = \mathbf{K}[\texttt{free\_dofs}, :][:, \texttt{free\_dofs}], \qquad \mathbf{M}_{\text{free}} = \mathbf{M}[\texttt{free\_dofs}, :][:, \texttt{free\_dofs}] \,.$$

Auf diese reduzierten Matrizen wird das verallgemeinerte Eigenwertproblem der Strukturdynamik angewendet.

2.2.2 Aufbau der Operatoren

Um die verschiedenen Lösungsvarianten des diskreten Eigenwertproblems zu untersuchen, wurden im Code drei Operatoren implementiert:

- **Konsistent ($\mathbf{K}_{\text{free}}^{-1}\mathbf{M}_{\text{free}}$):** Berechnung eines Operators
  $$\mathbf{A}_K = \mathbf{K}_{\text{free}}^{-1}\mathbf{M}_{\text{free}}$$
  mittels direkter Inversion `torch.linalg.inv(K_free)`.
- **Konsistent ($\mathbf{M}_{\text{free}}^{-1}\mathbf{K}_{\text{free}}$):** Analog wird
  $$\mathbf{A}_M = \mathbf{M}_{\text{free}}^{-1}\mathbf{K}_{\text{free}}$$
  mit `torch.linalg.inv(M_free)` gebildet.
- **Geklumpte Masse (Lumped):** Die Massenmatrix wird diagonalisiert, indem die Zeilensummen gebildet werden. Die Invertierung erfolgt elementweise:
  $$\mathbf{M}_{\text{lumped}}^{-1} = \text{diag}\left(\frac{1}{\sum_j M_{\text{free},ij}}\right),$$
  woraus der Operator $\mathbf{A}_{\text{L}} = \mathbf{M}_{\text{lumped}}^{-1}\mathbf{K}_{\text{free}}$ entsteht.

Die Eigenwerte dieser Operatoren werden anschließend mit `torch.linalg.eig` berechnet. Es werden nur reelle, positive Eigenwerte ausgewertet; daraus werden Kreisfrequenzen ω_i und Eigenfrequenzen in Hertz

$$f_i = \frac{\omega_i}{2\pi}$$

berechnet und nach aufsteigender Größe sortiert. Die ersten sieben Eigenfrequenzen der jeweiligen Variante werden im Terminal ausgegeben und in einem gemeinsamen Plot (`frequenzvergleich.png`) den analytischen Balkenfrequenzen gegenübergestellt. Die so erhaltenen Eigenfrequenzen und Eigenvektoren bilden die Datengrundlage für die Validierung und den Methodenvergleich in Kapitel 3.

Es ist zu beachten, dass die explizite Berechnung von Matrixinversen hier nur aufgrund der geringen Systemgröße (ca. 330 globale Freiheitsgrade vor Reduktion) angewandt wurde. Für große FE-Modelle wären stattdessen verallgemeinerte Eigenwertlöser sinnvoll, die ohne explizite Inversion auskommen.

2.3 Quadratische Ansatzfunktionen

Zur Verbesserung der Approximation wurde ein quadratisches Serendipity-Element (Q8) gewählt. Es besitzt vier Eckknoten sowie je einen Mittelknoten pro Elementseite und erlaubt gegenüber dem bilinearen Q4-Element eine deutlich genauere Abbildung gekrümmter Geometrien und größerer Gradienten, ohne den vollständigen Ansatzraum des neunknotigen Lagrange-Elements zu benötigen.

Die Formfunktionen gliedern sich in zwei Gruppen:

- **Eckknotenfunktionen**: an den zugehörigen Ecken gleich 1, auf gegenüberliegenden Kanten 0.
- **Seitenmittelknotenfunktionen**: nur auf einer Elementseite ungleich null, Maximum in der Seitenmitte.

Die Serendipity-Eigenschaft reduziert die Anzahl der Freiheitsgrade bei dennoch guter Konvergenz. Da die Formfunktionen Polynome bis Grad 2 enthalten, wird zur exakten Integration eine 3×3-Gauß-Quadratur auf dem Masterelement $[-1, 1] \times [-1, 1]$ verwendet. An allen neun Integrationspunkten werden die partiellen Ableitungen der Formfunktionen nach den natürlichen Koordinaten ξ und η analytisch bestimmt, um die Jacobi-Matrix sowie die Ableitungen der Verschiebungen im physikalischen Raum zu berechnen. Die quadratischen Ansatzfunktionen erhöhen den Rechenaufwand jedoch geringfügig.

2.4 Newmark-Beta-Verfahren

Für die transiente Analyse der Balkenstruktur wird das Newmark-Beta-Verfahren zur numerischen Zeitintegration der Bewegungsgleichung eingesetzt. Das Verfahren basiert auf einer diskreten Approximation der Kinematikgrößen Verschiebung, Geschwindigkeit und Beschleunigung über die Parameter β und γ. Je nach Wahl dieser Parameter kann das Verfahren sowohl implizite als auch explizite Integratoren abbilden. In der vorliegenden Implementierung wird die Struktur zunächst mithilfe der aus der Modalanalyse bekannten Massen- und Steifigkeitsmatrix in die Bewegungsgleichung

$$\mathbf{M\ddot{u}} + \mathbf{Ku} = \mathbf{f} \tag{2.1}$$

eingesetzt. Aus der Wahl der Zeitparameter ergibt sich ein Zeitschritt

$$\Delta t = \frac{T}{N}, \tag{2.2}$$

der über das Verhältnis von Gesamtzeitintervall und Anzahl der Integrationsschritte festgelegt wird.

Für Werte $\beta \geq 0.25$ wird das unbedingte stabile Verfahren der konstanten Durchschnittsbeschleunigung verwendet. Hierzu wird eine modifizierte Steifigkeitsmatrix

$$\mathbf{K}_{\text{eff}} = \mathbf{K} + \frac{1}{\beta \Delta t^2}\mathbf{M} \tag{2.3}$$

aufgebaut. In jedem Zeitschritt werden die neuen Zustandsgrößen aus einem linearen Gleichungssystem bestimmt, wodurch Stabilität auch bei größeren Zeitschritten gewährleistet ist.

Für $\beta < 0.25$ geht die Methode in ein explizites Verfahren über, das dem Central-Difference-Scheme entspricht. Hierbei entfällt die Lösung eines Gleichungssystems; die Verschiebung im nächsten Zeitschritt ergibt sich direkt aus einer Rekursionsformel. Das Verfahren ist jedoch nur bedingt stabil und erfordert die Einhaltung der klassischen Stabilitätsbedingung

$$\Delta t \leq \frac{2}{\omega_{\max}}, \tag{2.4}$$

wobei $\omega_{\max}$ die höchste Eigenkreisfrequenz der reduzierten Systemmatrizen ist. Die benötigten Eigenwerte werden dabei aus der zuvor durchgeführten Modalanalyse entnommen.

Beide Varianten werden im Code implementiert: Bei großen β-Werten wird das implizite, bei kleinen Werten das explizite Verfahren genutzt. Die resultierenden Verschiebungen der Struktur werden fortlaufend visualisiert, während zusätzlich die zeitliche Entwicklung der Verschiebung am freien Balkenende aufgezeichnet wird. Dies ermöglicht eine direkte Bewertung der Stabilität und des dynamischen Verhaltens der gewählten Parameterkombination.

3 Modalanalyse

Die von uns implementierte Modalanalyse zur Lösung des diskreten Eigenwertproblems wird durchgeführt, um die Blackbox der Finiten-Elemente-Methode zu öffnen und den grundlegenden Charakter der Schwingungseigenschaften besser zu verstehen. Die Modalanalyse kann auf unterschiedlichen Wegen durchgeführt werden. Diese unterscheiden sich grundsätzlich in dem Rechenaufwand und der resultierenden Genauigkeit. Daher sollte je nach Größe und Anforderungen an die Lösungsgenauigkeit entschieden werden, welche Lösungsstrategie zielführend ist.

3.1 Modalanalyse Validierung - Python vs. Ansys

Um unser FEM-Modell in Python hinsichtlich der Modalanalyse auszuwerten und zu überprüfen werden nachfolgend ermittelte Moden des Python-Skriptes mit Moden aus der Software ANSYS verglichen. Hierbei muss jedoch nicht nur die Eigenfrequenz berücksichtigt werden, sondern auch der Eigenvektor. Nachfolgend wird exemplarisch ein Vergleich angestellt. In der Tabelle 1 werden die ersten 10 berechneten Moden beziehungsweise Eigenfrequenzen aus dem Python-Skript mit denen aus ANSYS gegenüber gestellt. Es wurde folgendes Setup in Python wie ANSYS verwendet:

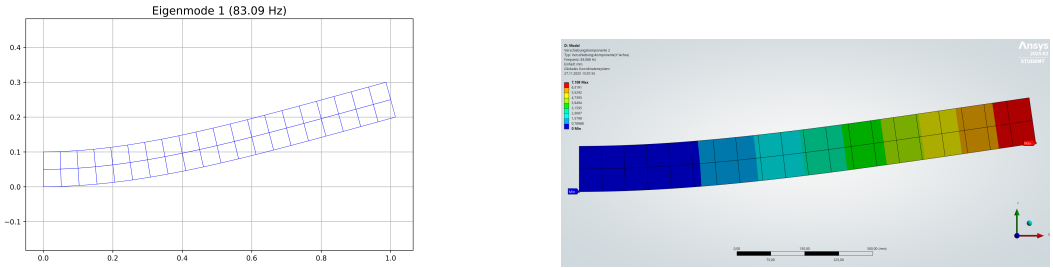
- Länge des Balkens: 1 m
- Höhe des Balkens: 0,1 m
- Anzahl der Elemente in X-Richtung: 20
- Anzahl der Elemente in Y-Richtung: 2
- Ansatzfunktion: quadratisch
- Materialparameter: Identisch wie in HA1

Einfache Systeme werden zumeist auf die ersten 20 Eigenmoden untersucht, da höhere Moden eine größere Frequenzanregung benötigen und im allgemeinen eine kleine Amplitude besitzen. Für dieses Beispiel reicht es jedoch die ersten zehn Eigenfrequenzen zu vergleichen.

Es ist zu erkennen, dass bei höheren Moden die berechneten Eigenfrequenzen von den ANSYS-Eigenfrequenzen abweichen. Der Fehler hält sich jedoch im einstelligen Prozent-Bereich und ist somit tolerierbar.
Nachfolgend werden nun die berechneten Moden mittels Visualisierung ausgewertet, um sicherzustellen, dass die Eigenvektoren auch vergleichbar sind. Hierzu einige ausgewählte Beispiele:

Mode	Eigenfrequenz Python [Hz]	Eigenfrequenz Ansys [Hz]	Relative Abweichung [%]
1	83,1	83,07	0,04
2	498,7	498,56	0,03
3	1294,9	1294,80	0,01
4	1314,4	1313,90	0,04
5	2390,4	2388,80	0,07
6	3648,8	3645,20	0,10
7	3881,9	3881,70	0,01
8	5029,1	5021,90	0,14
9	6456,0	6459,60	0,06
10	6492,0	6479,00	0,20
Mittel	-	-	**0,07**

Tab. 1: Vergleich der ersten zehn Eigenfrequenzen zwischen der Python-FEM und ANSYS (aktualisiert)

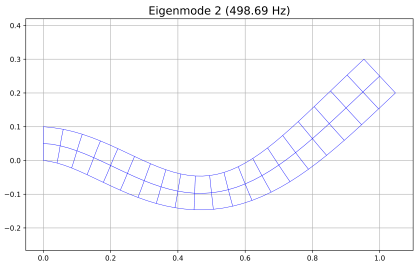


(a) Erste Eigenform Python

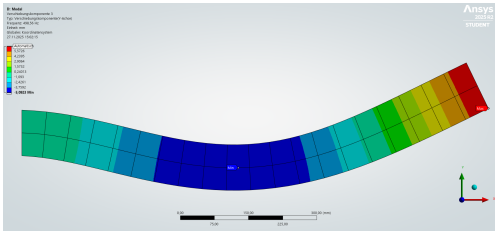
(b) Erste Eigenform ANSYS

Abb. 3: Vergleich der ersten Eigenform

Wie erkennbar, stimmen die jeweiligen Eigenmoden überein. Es wurde darauf geachtet, dass identische Material- und Vernetzungsparameter sowie Randbedingungen gewählt wurden. Wichtig bei einer solchen Visualisierung ist der Skalierfaktor, mit dem das Ergebnis skaliert wird, da sonst keine Verformung erkennbar ist. Die Ergebnisse aus ANSYS wurden mit einem Faktor von 15 skaliert.

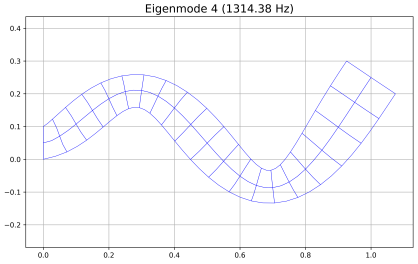


(a) Zweite Eigenform Python

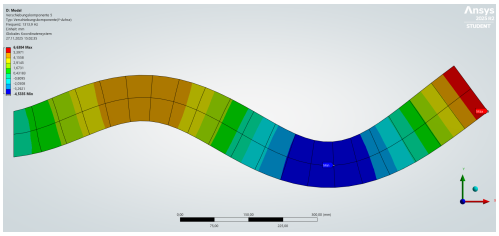


(b) Zweite Eigenform ANSYS

Abb. 4: Vergleich der zweiten Eigenform



(a) Vierte Eigenform Python



(b) Vierte Eigenform ANSYS

Abb. 5: Vergleich der vierten Eigenform

3.2 Vergleich der Verfahren $\mathbf{K_{tilde}}^{-1}, \mathbf{K_{free}}^{-1}, \mathbf{M_{free}}^{-1}, \mathbf{M}_{lumped}^{-1}$

Das Ziel aller Verfahren ist die Berechnung der Eigenfrequenzen (EF) und Eigenformen (EV).
Die genannten Lösungsstrategien des diskreten Eigenwertproblems (EWP) lassen sich grundlegend in 4 Kategorien einteilen:

- Berechnung mit Penalty-Steifigkeitsmatrix (Einträge mit Penalty Werten bleiben enthalten - Speicherfreundlich - Inversenunfreundlich)
- Berechnung mit reduzierter Steifigkeits- & Massenmatrix (Einträge mit Randbedingungen werden entfernt - Speicherunfreundlich - Inversenfreundlich)
- Berechnung mit Inverser Steifigkeitsmatrix vs. Berechnung mit Inverser Massenmatrix (linkes EWP vs. rechtes EWP - je nach Ïnversenaufwand")
- Berechnung mit invertierter lumped Massmatrix (für maximale Effizienz bei Inversenberechnung - auf Kosten der Genauigkeit v.a. bei höheren EF)

Die quantitativen Unterschiede der Ergebnisse dieser Methoden, werden anschaulich im Terminal des Programms 'HA2_quad_modal_Newmark.py' dargestellt. Des weiteren kann ein grafischer Vergleich der Methoden bei ansteigender Modenzahl in dem Plot 'Modale Frequenzen: Analytisch vs FEM' der Ausgabe des Programms 'JIN_nonlin_ha2_1.py' gefunden werden.

3.2.1 Handhabung von Instabilitäten durch Anpassung der Penalty-Methode

In der Penalty-Methode können zu hohe Werte des Penalty-Parameters zu numerischen Instabilitäten führen. Diese entstehen durch eine starke Ungleichheit der Matrizenwerte, was Rundungsfehler verstärkt und die Lösung des Systems erschwert. Solche Probleme treten besonders in großen Diskretisierungen auf, wo die Berechnung empfindlich auf kleine Abweichungen reagiert.

Ursprünglich wurde ein Wert von 10^{22} verwendet, der die Randbedingungen sehr streng durchsetzt, aber die Stabilität beeinträchtigt. Durch einen Wert von 10^{13} wird die Verteilung der Matrizenwerte ausgeglichener. Dadurch sinkt das Risiko von Fehlern bei der Auflösung, und die Simulation bleibt zuverlässig. Die Randbedingungen werden weiterhin gut erfüllt, mit einem akzeptablen Fehleranteil von etwa 10^{-13}.

Diese Anpassung verbessert die Robustheit der Methode, ohne die Genauigkeit wesentlich zu opfern. Sie eignet sich besonders für iterative Lösungsverfahren, die sonst langsam oder fehleranfällig wären. In der Praxis sollte der Parameter je nach Problemgröße getestet werden, um den besten Ausgleich zu finden.

3.2.2 Reduzierte Steifigkeitsmatrix – Funktionsweise, Vorteile und Nachteile

Funktionsweise Die reduzierte Steifigkeitsmatrix entsteht durch die Partitionierung des globalen Gleichungssystems. Das System

$$\mathbf{K}\mathbf{u} = \mathbf{F} \tag{3.1}$$

wird in freie (f = free) und durch Randbedingungen belegte Freiheitsgrade (c = constrained) unterteilt [3–5]:

$$\begin{bmatrix} \mathbf{K}_{ff} & \mathbf{K}_{fc} \\ \mathbf{K}_{cf} & \mathbf{K}_{cc} \end{bmatrix} \begin{bmatrix} \mathbf{u}_f \\ \mathbf{u}_c \end{bmatrix} = \begin{bmatrix} \mathbf{F}_f \\ \mathbf{F}_c \end{bmatrix}. \tag{3.2}$$

Für homogene Dirichlet-Randbedingungen gilt $\mathbf{u}_c = \mathbf{0}$. Durch Einsetzen in die erste Zeile reduziert sich das System zu

$$\mathbf{K}_{ff}\,\mathbf{u}_f = \mathbf{F}_f, \tag{3.3}$$

wobei $\mathbf{K}_{ff}$ die sogenannte reduzierte Steifigkeitsmatrix darstellt. Das Gleichungssystem muss somit nur noch für die unbekannten Verschiebungen $\mathbf{u}_f$ gelöst werden. Die zweite Zeile ($\mathbf{K}_{cf}\mathbf{u}_f + \dots$) dient anschließend lediglich zur Bestimmung der Lagerreaktionen $\mathbf{F}_c$.

Vorteile Im Vergleich zur Penalty-Methode bietet dieses Verfahren signifikante Vorzüge:

- **Numerische Exaktheit:** Da die Dirichlet-Randbedingungen durch Elimination exakt erfüllt werden ($\mathbf{u}_c \equiv \mathbf{0}$), entfällt der Approximationsfehler, der bei der Penalty-Methode durch endliche Penalty-Faktoren entsteht.
- **Numerische Stabilität (Konditionierung):** Die Penalty-Methode fügt sehr große Zahlenwerte in die Hauptdiagonale ein, was die Konditionszahl der Matrix verschlechtert (Ill-Conditioning"). Die reduzierte Matrix $\mathbf{K}_{ff}$ behält hingegen die natürliche Größenordnung der Steifigkeitseinträge bei, was Rundungsfehler minimiert.
- **Reduzierte Systemgröße:** Die Dimension des zu lösenden Gleichungssystems verringert sich ($N_{red} < N_{ges}$), was die Rechenzeit für den Solver, insbesondere bei Eigenwertproblemen und transienten Analysen, reduziert.

Nachteile Dem stehen implementierungstechnische Herausforderungen gegenüber:

- **Erhöhter Implementierungsaufwand (Index-Mapping):** Durch das Entfernen von Gleichungen geht die direkte Zuordnung zwischen Knotennummer und Matrix-Index verloren. Es ist ein Mapping-Algorithmus (z. B. Destination Arrays) erforderlich, um globale Freiheitsgrade auf das reduzierte System abzubilden und die Ergebnisse nach der Lösung wieder korrekt zuzuweisen.
- **Speicheroperationen (Matrix-Resizing):** Obwohl die finale Matrix kleiner ist, ist der Prozess der Reduktion rechenintensiv. Das nachträgliche Entfernen von Zeilen und Spalten aus einer assemblierten Matrix erfordert in kompilierten Sprachen (wie C++ oder Fortran) oft das Umkopieren von Speicherblöcken, was bei sehr großen Modellen ineffizient sein kann ("Memory Copy Overhead").

4 Transiente Analyse – Newmark-Beta-Verfahren

Als ein implizites Zeitintegrationsverfahren wurde das Newmark-Beta-Verfahren implemnetiert.

4.1 Simulation Kragbalken, freie Schwingung

Um die Zeitintegration mittels des Newmark-Beta-Verfahrens zu verifizieren, wurde die freie Schwingung des Kragbalkens simuliert. Grundlage ist das diskrete Bewegungsgleichungssystem

$$\mathbf{M}\ddot{\mathbf{u}}(t) + \mathbf{K}\,\mathbf{u}(t) = \mathbf{F}_{\text{ext}}(t), \tag{4.1}$$

wobei die konsistente Massenmatrix $\mathbf{M}$ und die Steifigkeitsmatrix $\mathbf{K}$ aus der zuvor beschriebenen Q8-Diskretisierung stammen.

4.1.1 Versuchsaufbau und Randbedingungen

Die Simulation erfolgt in zwei Phasen:

1. **Initiale Auslenkung ($t = 0$):** Als Anfangszustand $\mathbf{u}_0$ dient die statische Verformung unter einer transversalen Einzellast von $F = 1000\,\mathrm{N}$ am freien Ende. Das System startet aus der Ruhe, d. h. die Anfangsgeschwindigkeit beträgt $\dot{\mathbf{u}}_0 = \mathbf{0}$. Die zugehörige Anfangsbeschleunigung $\ddot{\mathbf{u}}_0$ wird konsistent aus der Gleichgewichtsbedingung

$$\mathbf{M}\,\ddot{\mathbf{u}}_0 = \mathbf{F}_{\mathrm{ext}}(0) - \mathbf{K}\,\mathbf{u}_0$$

   bestimmt.

2. **Freie Schwingung ($t > 0$):** Unmittelbar für alle Zeitschritte $t > 0$ wird die äußere Last entfernt ($\mathbf{F}_{\mathrm{ext}} = \mathbf{0}$). Dadurch geht das System in eine freie, ungedämpfte Schwingung über.

Für die Zeitintegration wurde das implizite Newmark-Beta-Verfahren mit den Parametern

$$\beta = 0{,}25 \quad \text{und} \quad \gamma = 0{,}5$$

gewählt (Methode der mittleren konstanten Beschleunigung). Diese Parameterwahl garantiert bedingungslose Stabilität und Energieerhaltung für lineare elastische Systeme.

Die Simulationsdauer beträgt $t_{\mathrm{end}} = 0{,}01\,\mathrm{s}$ bei $N = 100$ Zeitschritten, was einer Zeitschrittweite von $\Delta t = 10^{-4}\,\mathrm{s}$ entspricht. Dies gewährleistet eine hohe zeitliche Auflösung der ersten Eigenmode ($f_1 \approx 83\,\mathrm{Hz}$), da $\Delta t \ll T_1$ gilt.

4.1.2 Ergebnisse und Validierung

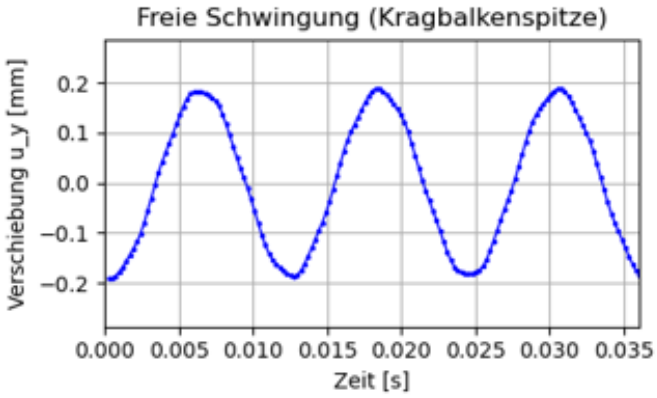


Abb. 6: Zeitlicher Verlauf der vertikalen Verschiebung u_y am freien Ende des Kragbalkens.

Abbildung 6 zeigt den zeitlichen Verlauf der vertikalen Verschiebung u_y an der Balkenspitze. Folgende Merkmale bestätigen die korrekte Durchführung der Simulation:

- **Startwert:** Die Schwingung beginnt bei der statischen Auslenkung von ca. 0,19 mm, was der analytischen Erwartung entspricht.
- **Energieerhaltung:** Die Amplitude der Schwingung bleibt über den gesamten Zeitraum konstant. Dies bestätigt, dass das implementierte Verfahren keine numerische Dämpfung einführt ($\gamma = 0{,}5$) und das System korrekt als ungedämpft modelliert wurde.
- **Frequenz-Konsistenz:** Der Antwortverlauf entspricht einer harmonischen Schwingung. Die aus dem Plot ablesbare Periodendauer $T \approx 0{,}012\,\mathrm{s}$ korrespondiert mit einer Frequenz von $f \approx 83{,}3\,\mathrm{Hz}$. Dieser Wert stimmt hervorragend mit der in der Modalanalyse ermittelten ersten Eigenfrequenz ($f_1 = 83{,}09\,\mathrm{Hz}$) überein.

4.2 Parameterstudie - Instabilität

Mit die häufigste Ursache für Instabilitäten bei der Transienten Analyse ist die Parameterwahl von β und γ. Bei konstantem $\gamma = 0,5$ und einer Wahl für $\beta = 0$ reduziert sich das implizite Verfahren und es liegt außerhalb des unbedingt stabilen Bereichs. Durch Testen konnten erste Instabilitäten für $\beta < 0,25$ erkannt werden. Demnach ist das Verfahren nur für Werte größer gleich 0,25 stabil. Des weiteren ist die Wahl der Zeitschrittgröße Δt von Interesse. Bei bedingt stabilen Parametern führen zu große Zeitschritte zu Instabilität. Die beiden beschrieben Instabilitäten werden in Abschnitt 4.3 genauer beschrieben. Weiterhin sind implizite Verfahren für lineares Verhalten geeignet,

zeigen jedoch bei starken Nichtlinearitäten instabiles Verhalten. Zuletzt wäre noch ein instabiles Verhalten bei unzureichender Diskretisierung zu erwähnen.

4.2.1 Bestimmen der Zeitschrittweite und Verhaltensanalyse

Es soll exemplarisch und durch Experimentieren mit den Parametern $\gamma = 0,5$ und $\beta = 0$ die kritische Zeitschrittweite bestimmt werden, ab welcher es zu Instabilitäten kommen kann. Rechnerisch ergibt sich die kritische Zeitschrittweite aus [6]

$$\Delta t \leq \frac{2}{\omega_{max}}. \tag{4.2}$$

Im Programm kann die Schrittweite durch Anpassung der steps und des zu betrachteneden time-intervall angepasst werden. Die Schrittweite ergibt sich aus dem Quotienten von time-intervall und steps. Da bei dem expliziten Verfahren extrem hohe Eigenwerte auftreten können, kann die kritische Schrittweite sehr klein werden. Es ergibt sich für einen maximalen Eigenwert von circa $8{,}8 \cdot 10^5$ eine minimale kritische Schrittweite von ungefährt $2{,}2 \cdot 10^{-6}$ s. Dieser Eigenwert ist abnhängig von der Konfiguration des Balkens, wird jedoch im Terminal ausgegeben hinsichtlich der Nachvollziehbarkeit.
Allgemein besagt das Stabilitätskriterium, dass bei Nichterfüllung ein stabiles Verhalten der Berechnung nicht garantiert werden kann. Für besonders hohe Abweichungen von der Stabilitätsbedingung zeigt sich eine deutlich schnellere Instabilität beziehungsweise Zerfall der Berechnung bei dem Skript. Die Auslenkung am Balkenende werden sprunghaft und die Elemente beginnen sich zu verzerren.
Das Verhalten bei der resultierenden expliziten Analyse weist gemäß der Stabilitätsbedingung eine deutlich kleinere Schrittweite im Vergleich zu der impliziten Analyse für eine stabile Berechnung auf. Es zeigt sich eine quasi verlangsamte Variante der impliziten Methode. Aufgrund der deutlich kleineren Schrittweite wirkt die Darstellung wie in slow-motion. Das Ergebnis bleibt identisch zu dem der impliziten Berechnung. Der Rechenaufwand und die damit verbundene Zeit ist jedoch deutlich höher.

4.3 Vergleich für ausgewählte Parameter

Zuletzt werden die Ergebnisse für ein konstantes $\gamma = 0,5$ mit $\beta_1 = 0$ und $\beta_2 = 0,25$ verglichen und diskutiert. Allgemein lassen sich anhand der gewählten Parameter die beiden Fälle wie folgt einordnen:

1. für β_1: zentrale Differenzquotienten (Central Difference Scheme) [7]
2. für β_2: konstante Durchschnittsbeschleunigungsverfahren [7]

Durch die Wahl von $\beta_1 = 0$ reduziert sich das Central Difference Scheme zu einem expliziten Verfahren. Hinsichtlich der Stabilität unterscheiden sich die Verfahren. Das konstante Durchschnittbeschleunigungsverfahren ist unbedingt stabil ([7]) und bedarf keiner Zeitschrittlängenbeschränkung nur aus Stabilitätsgründen. Im Gegensatz dazu ist das Central Difference Scheme nur bedingt stabil und bedarf einer Stabilitätsbedingung, welche lautet [6]

$$\Delta t \leq \frac{2}{\omega_{max}}. \tag{4.3}$$

Das bedeutet, dass je feiner diskretisiert wird, desto höher werden die Eigenfrequenzen und folglich kleiner die Zeitschrittlänge Δt, um die Stabilitätsbedingung zu erfüllen. Beide Verfahren haben ihre eigenen Vor- und Nachteile, welche bei der Parameterauswahl berücksichtigt werden müssen. So eignet sich das Central Difference Scheme gut für schnelle Verfahren mit hochfrequenten oder stark nichtlinearen Problemen, wie zum Beispiel komplexe Kontaktprobleme. Jedoch muss die Stabilitätsbedingung beachtet werden. Das konstante Durchschnittsbeschleunigungsverfahren eignet sich gut für stabile und langzeitige Simulationen, bei denen große Zeitschritte verwendet werden. Es ist jedoch nur auf lineare Probleme und einfache Kontaktprobleme anwendbar.

4.4 Numerische Dämpfung

Numerische Dämpfung (Dissipation) bezeichnet die Amplitudenreduktion oszillierender Komponenten in diskretisierten Lösungen steifer Differentialgleichungssysteme. Sie stabilisiert Simulationen, indem hochfrequente Störungen (jenseits der Nyquist-Frequenz $\pi/\Delta t$) abgeschwächt werden, während niederfrequente Moden erhalten bleiben. Bei $t \to \infty$ führt sie zu einer asymptotischen Abnahme der Amplitude, was in impliziten Methoden wie Newmark-β für mechanische Systeme relevant ist.

Die Stabilitätsfunktion $R(z)$ bestimmt den Spektralradius $\rho(\chi) = |R(i\chi)|$. Bei $\beta = 0,25$, $\gamma = 0,5$ gilt $\rho = 1$ (energieerhaltend) wie in 4.1.2 erwähnt. Asymptotisch ergibt sich $\rho(\infty) = |(\gamma - \beta)/\beta| < 1$ für $\gamma > 1/2$ (L-Stabilität).

In FEM-Implementierungen entsteht die Abnahme implizit durch β, γ und Δt. In freien Schwingungen oszilliert die Amplitude gedämpft ab. [8, S. 39, 55]

5 Ausblick

Zukünftige Arbeitspakete könnten es Optimierung des erarbeiteten Skriptes sein. Es zeigt sich, dass eine Speicherung der Matrizen im Sparse-Format durchaus geeignet wäre, da damit komplexere beziehungsweise feinere Diskretisierungen berechnet werden können, ohne das der Speicher überläuft und es zu einem Abbruch der Berechnung kommt. Zudem erlaubt Sparse-Arithmetik einen deutlich effizienteren Aufbau der globalen Matrizen sowie schnellere Matrix-Vektor-Produkte in der Dynamik.
Zusätzlich können die Eigenwerte direkt mittel bereits definierter Funktionen berechnet werden, wie sie in numerisch Bibliotheken (`scipy.sparse.linalg.eigs`, `torch.linalg.eigh`) verfügbar sind, wodurch das explizite Invertieren der Matrizen entfällt und sowohl Genauigkeit als auch Stabilität der Modalanalyse verbessert wird, ohne dass Invertierungen erfolgen müssen.

Literaturverzeichnis

[1] ETH Zürich. Introduction to Finite Element Analysis. URL: https://ethz.ch/content/dam/ethz/special-interest/mavt/mechanical-systems/mm-dam/documents/Notes/IntroToFEA_red.pdf.

[2] Roland Gomeringer u. a. Tabellenbuch Metall. ger. 49., neu bearbeitete und erweiterte Auflage, korrigierter Nachdruck 2024. Europa-Fachbuchreihe für Metallberufe. Haan-Gruiten: Verlag Europa-Lehrmittel Nourney, Vollmer GmbH & Co. KG, 2024. ISBN: 978-3-7585-1142-4.

[3] Olgierd Cecil Zienkiewicz, Robert Lee Taylor und J. Z. Zhu. The finite element method: its basis and fundamentals. en. 6. Aufl. Burlington (Mass.): Elsevier/Butterworth-Heinemann, 2005. ISBN: 978-0-7506-6320-5.

[4] Klaus-Jürgen Bathe. Finite element procedures. en. Englewood Cliffs (N. J.): Prentice Hall, 1996. ISBN: 978-0-13-301458-7.

[5] Thomas J. R. Hughes. The finite element method: linear static and dynamic finite element analysis. en. 1. Dr. Englewood Cliffs, N.J.: Prentice Hall, 1987. ISBN: 978-0-13-317025-2 978-0-13-317017-7.

[6] Nathan M. Newmark und Nathan M. Newmark. „A method of computation for structural dynamics“. en. In: Journal of the Engineering Mechanics Division 85 (EM3).3 (1959), S. 67–94. DOI: 10.1061/JMCEA3.0000098.

[7] 3.2.6.6. Newmark method — OpenSees documentation. en. 2022. URL: https://opensees.github.io/OpenSeesDocumentation/user/manual/analysis/integrator/Newmark.html?utm_source=chatgpt.com (besucht am 20. Nov. 2025).

[8] Jacob Fish und Wen Chen. „On accuracy, stability and efficiency of the newmark method with incomplete solution by multilevel methods“. en. In: International Journal for Numerical Methods in Engineering 46.2 (1999). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/%28SICI%291097-0207%2819990920%2946%3A2%3C253%3A%3AAID-NME673%3E3.0.CO%3B2-9, S. 253–273. ISSN: 1097-0207. DOI: 10.1002/(SICI)1097-0207(19990920)46:2<253::AID-NME673>3.0.CO;2-9. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-0207%2819990920%2946%3A2%3C253%3A%3AAID-NME673%3E3.0.CO%3B2-9 (besucht am 23. Nov. 2025).