Dokumentation SIRIUS

Teil3: Programmbeschreibung

Autor: Roland Wegmann

Stand: 27.04.2017

Inhaltsverzeichnis

3 Programmbeschreibung

3.1 Einführung

3.1.1 Aufgabe der Programmbeschreibung

Neben der Beschreibung der physikalischen Grundlagen (Teil 2) und der eigentlichen Bedienanleitung (Teil 4) soll die Programmbeschreibung den systematischen Aufbau, die "Philosophie" und die internen numerischen und programmtechnischen Verfahren des Programms SIRIUS ausführlich beschreiben. Sie soll dem interessierten Nutzer helfen, die Funktionsweise des Programms zu verstehen und damit eine Grundlage bilden für eine effektive Arbeit mit dem Programm.

Diese Beschreibung stellt neben den kommentierten Quelltexten des Programms auch eine unverzichtbare Informationsquelle für alle dar, die eventuell beabsichtigen, das Programm SIRIUS weiter zu entwickeln.

Die Programmbeschreibung soll deshalb die verwendeten und für den speziellen Einsatzfall angepassten und erweiterten numerischen Verfahren dokumentieren, die eventuell auch für die rechentechnische Lösung anderer Probleme von Interesse sein können. Der Autor des Programms möchte damit auch seine Erfahrungen auf diesem Gebiet möglichst umfassend weitergeben.

3.1.2 Was das Programm kann

Das Programm simuliert die Schmierung in **Radialgleitlagern**.

Grundlage der Berechnungen sind die **erweiterte Reynoldssche Differentialgleichung** nach Wegmann [20], bei der die Schmiermittelströmung im gesamten Schmierspalt abgebildet wird, einschließlich der Kavitation im sich erweiternden Schmierspalt oder die **klassische hydrodynamische Schmiertheorie** mit der Reynoldsschen Differentialgleichung in Verbindung mit den Gümbelschen Randbedingungen. Siehe dazu auch Abschnitt 2.1.3.

Es können **hydrodynamische und hydrostatische Lager** einschließlich Mischformen simuliert werden.

Es können **Schmiertaschen in beliebiger Anzahl, Form und Anordnung** dargestellt werden.

Es können **stationäre und instationäre Betriebsbedingungen** untersucht werden.

Aus einem vorgegebenen Lastverlauf über die Zeit kann die Verlagerung der Welle innerhalb der Lagerschale über die Zeit berechnet werden (**Verlagerungsbahn**) und so die sich einstellende minimale Schmierspalthöhe ermittelt werden. Es kann aber auch aus einer gemessenen oder vorgegebenen Verlagerungsbahn der Welle innerhalb der Lagerschale die Lagerbelastung über die Zeit in Größe und Richtung ermittelt werden.

Neben dem achsparallelen Lager mit ideal zylindrischer Welle und Lagerschale können auch Lage- und Formabweichungen berücksichtigt werden. Die Daten für diese Abweichungen können durch einfache Funktionen mit wenigen Parametern für grundsätzliche Untersuchungen modelliert werden oder durch punktweise gegebene Datenreihen aus Messungen oder vorgelagerten Berechnungen gegeben sein.

Die möglichen Lage- und Formabweichungen sind:

Eine **verkantete Welle** innerhalb der Lagerschale. Die Verkantung kann zeitlich variabel sein.

Eine durch ein Biegemoment **gebogene Welle**. Die Biegung kann zeitlich variabel sein.

Eine von der ideal zylindrischen Form abweichende Welle und/oder Lagerschale. Diese **Formabweichungen** werden als konstant über die Zeit angenommen.

Es können sowohl Lagerschalen dargestellt werden, die die Welle vollständig umschließen (**voll umschlossene Lager**) als auch Lagerschalen, die als Gleitschuhe die Welle nur teilweise umschließen (**teilweise umschlossene Lager**).

Es können **variable Drehzahlen** einschließlich Pendelbewegungen der Welle dargestellt werden.

Als Sonderfall kann ein Lager mit **radial versetzten axialen Lagerabschnitten** simuliert werden, welches bei pendelnder Bewegung eine erhöhte Tragfähigkeit gegenüber einem einfachen zylindrischen Gleitgelenk aufweist. Beispiele solcher Berechnungen sind in [34] dargestellt.

Es kann ein komplexes peripheres **hydrostatisches und/oder hydrodynamisches Schmiermittelversorgungssystem** modelliert werden mit mehreren Schmiermittelpumpen und verschiedenen Steuer- bzw. Regeleinrichtungen zur Verteilung des Schmiermittels auf die verschiedenen Schmiertaschen.

Als primäre Ergebnisse der Simulation der Schmiermittelströmung im Schmierspalt werden die **Schmiermitteldrücke im Schmierspalt** an jedem Gitterpunkt und zu jedem Zeitpunkt berechnet, sowie die **Drücke und Ölströme in dem peripheren Schmiermittelversorgungssystem**, soweit ein solches vorhanden ist und es wird die Wellenverlagerung bzw. die Lagerbelastung berechnet.

Als sekundäre Ergebnisse können daraus eine Reihe weiterer Ergebnisse berechnet und angezeigt werden. Das sind:

Die **Ölströme durch die einzelnen Schmiertaschen** in das Lager und über den Schmierspaltrand.

Die **Drücke, Ölströme und Leistungen einer hydrostatischen Schmiermittel-Versorgungseinrichtung**.

Die **Reibleistung**, die **durch die Wellenrotation** auf den Schmierfilm übertragen wird.

Die **innere Reibleistung**, die durch die dynamische Viskosität der angenommenen Newtonschen Schmierflüssigkeit im Schmierspalt in Wärme umgewandelt wird.

Die **Schmiermittelverteilung im Schmierspalt** unter Berücksichtigung der Kavitation im Schmierspalt.

Das **Gesamtvolumen des Schmierspalts** und das Gesamtvolumen der darin enthaltenen Schmierflüssigkeit.

Vom Lager aufgenommene **Kippmomente der Welle**, die aus einer Verkantung der Welle im Lager oder anderen asymmetrischen Bedingungen im Lager resultieren.

Die Datenein- und -ausgabe kann mit **dimensionslosen Daten** auf der Basis der Definition der Sommerfeldzahl oder mit **dimensionsbehafteten Daten** arbeiten. Dabei kann auch problemlos ein konkretes Lager in ein anderes geometrisch und physikalisch ähnliches Modelllager umgerechnet werden.

Die meisten **Eingabe- und Ergebnisdaten** können sowohl numerisch als auch in Zusammenarbeit mit der freien Grafiksoftware GNUPLOT [34] **grafisch dargestellt** werden. In Zusammenarbeit mit einem einfachen Filmschnittprogramm können für berechnete zeitliche Abläufe Animationen erzeugt werden. Es ist eine große Anzahl verschiedener Diagramme zur Ausgabe bereits vorbereitet. Diese Diagrammvorlagen können auch leicht den Wünschen des Anwenders angepasst und erweitert werden.

3.1.3 Was das Programm nicht kann

Es werden **keine Axiallager** berechnet.

Es werden **keine Kugelgelenke** berechnet.

Es wird mit einer konstanten dynamischen Viskosität im Schmierspalt gerechnet. Das bedeutet, dass im gesamten Schmierspalt eine konstante Temperatur angenommen wird. Deshalb können auch **keine Temperaturfelder** im Schmierspalt berücksichtigt werden.

Es kann **keine Festkörperreibung bzw. Mischreibung** im Lager simuliert werden. Allerdings kann mit Hilfe der ermittelten minimalen Spalthöhen im Vergleich mit den Rauhtiefen der Wellen- und Lagerschalenoberflächen festgestellt werden, ob im Lager mit Festkörperkontakt zu rechnen ist und wenn ja, bei welchen Betriebsbedingungen.

Es können **keine Kippsegmentlager** simuliert werden. (Mehrgleitflächenlager mit festen Gleitschuhen können durch eine punktweise Eingabe der speziellen Geometrie der Lagerschale abgebildet werden oder vereinfacht durch eine wellige Lagerschalenoberfläche)

Die Berücksichtigung zeitlich variabler **elastischer Verformungen** der Lager infolge der Schmierfilmdrücke im Lager ist noch in Arbeit [17] und deshalb in anwendungsreifer Form noch nicht möglich. Die programmtechnischen Voraussetzungen sind bereits implementiert, aber bisher nur teilweise dokumentiert. Der Algorithmus zur parallelen iterativen Berechnung von Verformung und Schmiermittel-Druckverteilung bedarf noch einer Weiterentwicklung, da er noch nicht zuverlässig zu stabilen Ergebnissen führt. Die Berücksichtigung stationärer elastischer Formabweichungen kann schon durch die punktweise Eingabe von Formabweichungen der Lagerschale, die in einem FEM-Programm berechnet wurden, berücksichtigt werden.

3.1.4 "Programmphilosophie"

Im Rahmen des BMWi-geförderten Projekts HYDROS [3], [22] in dem ein hydrostatisch-hydrodynamisches Hybridlager entwickelt wurde, wurde das Programm SIRIUS grundlegend überarbeitet und erweitert. Bei dieser Überarbeitung wurden folgende Ziele angestrebt: Das Programm sollte kein Ein-Zweck-Programm werden, sondern auch nach dem Projekt weiterhin für andere Anwendungsfälle geeignet sein, sowohl für wissenschaftliche Untersuchungen als auch für ingenieurtechnische Anwendungen. Trotz seiner wachsenden Komplexität soll es anwenderfreundlich sein. Es soll weiterentwicklungsfähig und für die Integration spezieller Anwendungen offen sein. Dabei mussten Kompromisse gefunden werden zwischen einer maximalen Erfüllung der oben benannten Ziele und der begrenzten verfügbaren Entwicklungskapazität. Die gewonnenen und eingearbeiteten Erkenntnisse sollen der wissenschaftlich und technisch interessierten Öffentlichkeit zur Verfügung gestellt werden und so einen Betrag zum Fortschritt in der hydrodynamischen Schmiertheorie leisten.

Daraus ergibt sich eine Programmphilosophie, die in den nachfolgenden Unterabschnitten dargestellt ist.

3.1.4.1 Breites Anwendungsspektrum

Die realisierte Breite des Anwendungsspektrums ist im Abschnitt 3.1.2 bereits aufgelistet.

3.1.4.2 Anwenderfreundlichkeit

Zum einfachen ersten Einstieg in das Programm werden die Daten für ein einfaches Demonstrationsbeispiel mit dem Start des Programms automatisch aufgerufen, so dass der neue Nutzer ohne weitere Dateneingabe eine erste Berechnung ausführen kann und so einen ersten Einblick in die Arbeitsweise erhält (siehe Schnelleinstieg Abschnitt 1.4). Dabei werden mit dem Start des Programms auch alle anderen potentiellen Eingabedaten mit einem Anfangswert belegt. Dieser Anfangszustand des Programms kann jederzeit wieder hergestellt werden.

Der Nutzer wird in einer linearen Abfolge von Eingabemenüs durch das Programm geführt, so dass am Ende eine vollständige Dateneingabe gewährleistet ist. Dabei unterliegt die Reihenfolge der Dateneingabe einer gewissen Hierarchie. So bestimmen die ersten Eingaben, welche nachfolgenden Eingaben noch erforderlich sind. Diese Führung durch das Programm ist aber kein Zwangsweg in eine Richtung. Es kann von fast jedem Punkt zurückgesprungen werden, um vorhergehende Eingaben zu korrigieren.

Aus der Vielzahl der im Programm vorgehaltenen Eingabemöglichkeiten werden nur die abgefragt, die für den gewählten Fall relevant sind. Es werden Angaben zum Definitionsbereich der Eingabedaten gemacht. Während der Eingabe erfolgt eine Plausibilitätsprüfung der eingegebenen Daten. Bei Eingabefehlern erfolgt eine Fehlermeldung mit der Aufforderung zur erneuten Eingabe. An einigen Stellen werden Warnungen ausgegeben, wenn eine Fehleingabe vermutet werden kann, aber nicht zwingend ist. So lange die Eingabe eines Wertes nicht abgeschlossen ist, kann die Eingabe abgebrochen werden, ohne dass der bisherige Wert geändert wird.

Neben der Direkteingabe der Daten über die Tastatur kann die Eingabe großer Datenmengen, die in Feldern abgespeichert sind, auch über das Einlesen von Textdateien (ASCII-Dateien) organisiert werden. Damit ist eine Schnittstelle zu anderen Programmen gegeben. Diese Textdateien sind mit einfachen Editoren lesbar, so dass bei Fehlersuche der Datenfluss leicht kontrolliert werden kann.

Die primären Eingabe- und Ergebnisdaten werden gemeinsam in einer Textdatei gesichert und können zur späteren Auswertung oder erneuten Bearbeitung wieder eingelesen werden. Es können auch nur teilweise bearbeitete Eingabedatensätze oder nur teilweise erfolgte Berechnungen für eine spätere weitere Bearbeitung abgespeichert werden.

Für die Berechnung konkreter Lager und/oder für prinzipielle Untersuchungen kann zwischen einer dimensionsbehafteten und einer dimensionslosen Datenein- und -ausgabe gewählt werden. Die Definition der dimensionslosen Daten ist passend zur Definition der Sommerfeldzahl eingerichtet. Alle Daten werden mit den 5 Bezugsparametern Wellendurchmesser d, relative Lagerbreite B, relatives Lagerspiel S, dynamische Viskosität η und Bezugswinkelgeschwindigkeit ωb dimensionslos gemacht. Sofern die 5 Bezugsparameter eingegeben sind, kann während der Dateneingabe im PreProzessor und während der Auswertung der Ergebnisse im PostProzessor problemlos zwischen dimensionsloser und dimensionsbehafteter Darstellung gewechselt werden. Intern arbeitet das Programm ausschließlich mit den dimensionslosen Daten, was die Programmierung erleichtert.

Während des Projekts HYDROS wurden mit SIRIUS zur Lösung der eigentlichen Entwicklungsaufgabe des Projekts viele Berechnungen durch geführt und damit das Programm ausführlich getestet und praktische Erfahrungen für die Anwendung des Programms gesammelt. Aus diesen Erfahrungen resultiert eine große Anzahl kleiner Routinen im Hintergrund, die die Arbeit mit dem Programm erleichtern und beschleunigen.

Die alte Programmiersprache FORTRAN 77, mit ihrem überschaubaren Befehlssatz, wurde bei der Überarbeitung des Quelltextes beibehalten. Damit ist es möglich, kompakten schnellen Programmcode zu erzeugen. So kann das Programm weiterhin auf üblichen Personalcomputern ausgeführt werden und die Berechnungszeiten sind meist so kurz, dass im direkten Dialog eine Vielzahl von Varianten durchgerechnet werden können, so dass ein virtuelles Experimentieren bei der Suche nach optimalen Lösungen möglich ist.

Bei der Auslegung eines hydrodynamisch geschmierten Gleitlagers ist letztendlich für den Nachweis seiner Funktionsfähigkeit ausschlaggeben, wie groß die minimale Schmierspalthöhe im Vergleich zu den Rauhtiefen der Gleitflächen ist. Neben der Berechnung der Schmierspalthöhe wird im Programm SIRIUS aber Wert darauf gelegt, das möglichst viele weitere Daten ermittelt und sowohl numerisch als auch grafisch dargestellt werden können. Damit soll sowohl der wissenschaftlich als auch ingenieurmäßig arbeitende Nutzer des Programms umfassenden Einblick in das Geschehen im Lager erhalten, was ihm helfen soll, die Prozesse im Lager zu verstehen und so zu einer optimalen Lösung zu kommen. Für die Interpretation dieser Ergebnisse sollte der Anwender des Programms bereits einige grundlegende Kenntnisse der hydrodynamischen Schmiertheorie besitzen.

3.1.4.3 Entwicklungsfähigkeit

Das Programm wurde von Anfang an als ein Baukastensystem konzipiert, indem es konsequent nach funktionellen Gesichtspunkten in Routinen aufgeteilt ist. So konnten in der Vergangenheit von dem ursprünglichen Programm für verschiedene Aufgabenstellungen verschiedene Varianten abgeleitet werden. Im Rahmen der Programmüberarbeitung wurde ein Großteil dieser Varianten in der überarbeiteten Version vereinigt und weiterentwickelt, was die mögliche Variantenvielfalt weiter erhöht hat.

Zur Beherrschung der Variantenvielfalt wurde ein umfangreiches System von Steuerparametern entwickelt, durch welches die Beschreibung der jeweils gewählten Lagervariante codiert wird. Diese Steuerparameter steuern sowohl die Abfrage der relevanten Eingabedaten, die Konsistenzprüfung des Datensatzes als auch die eigentliche Berechnung. Dieses System ist erweiterbar, indem innerhalb der vorhandenen Steuerparameter weitere Varianten hinzugefügt werden können oder auch neue Steuerparameter für einen zusätzlichen Variantenblock ergänzt werden. Die Codierung dieser Steuerparameter ist in der Beschreibung des Hauptmenüs " Festlegungen zur Theorie, zum Berechnungsverlauf und zum Lagertyp " in der Bedienanleitung Abschnitt 4.4.2 dokumentiert.

Auch für die große Zahl von Daten, die die physikalisch-technischen Zustände im Lager abbilden, wurde ein umfangreiches Parametersystem entwickelt. Diese Parameter wurden nach funktionellen Gesichtspunkten auf mehrere Common-Blöcke aufgeteilt. Auf diese können die verschiedenen Routinen in der Regel direkt zugreifen. Die Nomenklatur dieser Parameter wird in allen Routinen einheitlich gehandhabt.

Die alte Programmiersprache FORTRAN 77, mit ihrem überschaubaren Befehlssatz, ist leicht erlernbar. Sie erfordert aber einen sauberen Programmierstil, da sie über relativ wenig Kontrollmechanismen verfügt.

3.1.4.4 Kompromiss

Aus den oben erwähnten Gründen wurde die Programmiersprache FORTRAN durchgängig für die gesamte Programmierung verwendet. FORTRAN kennt aber noch keine Befehle für grafische Ein- und Ausgaben. Deshalb musste auf eine grafische Programmoberfläche verzichtet werden. Die textbasierte Programmoberfläche erscheint in dem spröden Charme des altbekannten DOS-Fensters, was für die jüngeren Anwender sicher etwas gewöhnungsbedürftig ist. Damit fehlen zwar einige Gestaltungsmöglichkeiten für eine kontrastreiche Programmoberfläche mit diversen Eingabefenstern. Erstaunlicherweise sind aber auch mit FORTRAN alle wesentlichen Anforderungen für einen professionellen Dialog zwischen Anwender und Programm gegeben. Durch die Gestaltung der Textblöcke der verschiedenen Menüs konnten auch gewisse grafische Effekte erzeugt werden, die nach einer Einarbeitungszeit eine schnelle Orientierung auf der Programmoberfläche ermöglichen. Auf die grafische Darstellung der Ergebnisse braucht trotzdem nicht verzichtet werden. Das kann mit dem eigenständigen Programm GNUPLOT im Parallelbetrieb gut realisiert werden.

3.1.4.5 SIRIUS als wissenschaftliche Informationsquelle

Im Programm SIRIUS sind ein reicher Erfahrungsschatz zur Theorie der hydrodynamischen Schmierung und Methoden ihrer Berechnung eingeflossen. Damit dieser Erfahrungsschatz nicht verloren geht, soll auf eine Kommerzialisierung des Programms verzichtet werden. So entsteht nicht die Versuchung, fachliches und methodisches Know-how als kommerziell verwertbares Betriebsgeheimnis zurück zuhalten. Deshalb sollen mit dieser ausführlichen Dokumentation und der Veröffentlichung des gesamten kommentierten Quelltextes möglichst viele Details für die Nutzung und Weiterentwicklung der hydrodynamischen Schmiertheorie offengelegt werden. Die ausführliche Beschreibung der technisch-physikalischen Grundlagen im Abschnitt 2 der Dokumentation kann dementsprechend als umfangreiche Formelsammlung zur Problematik des hydrodynamischen Gleitlagers genutzt werden.

Indem das Programm nur kostenfreie Hilfsprogramme verwendet und das Programm selbst mit einer kostenlos nutzbaren GNU-Lizenz versehen ist, steht es auch allen Studierenden zur freien Verfügung, um sich anhand konkreter Beispiele mit der Schmiertheorie vertraut zu machen.

3.2 Übersicht über die Programmstruktur

3.2.1 Das Gesamtsystem der Berechnung, Datensicherung und -übertragung und der grafischen Ausgabe

Das eigentliche Programm SIRIUS besteht in seiner kompilierten Form lediglich aus der Datei "SIRIUS.exe", mit einer Dateigröße von ca. 1,5 MB. Das Programm ist in dieser Form ohne weitere Hilfsmittel auf einem Windows-Betriebssystem lauffähig und kann so, ohne sonstige weitere Vorbereitungen oder Installationen, getestet werden, um einen ersten Eindruck seiner Funktionsweise zu gewinnen.

Intern ist SIRIUS in die Funktionseinheiten PreProzessor, Solver und PostProzessor gegliedert. Der PreProzessor dient der Dateneingabe. Im Hintergrund erfolgt dabei eine Konsistenzprüfung der eingegebenen "primären" Eingabedaten und eine Komplettierung des Eingabedatensatzes durch weitere "sekundäre" Eingabedaten, die sich aus den primären Eingabedaten berechnen lassen. Im Solver erfolgt die zeitintensive Hauptrechnung, die die "primären" Ergebnisdaten liefert. Im PostProzessor erfolgt die Sicherung der primären Ergebnisdaten gemeinsam mit den primären Eingabedaten und die Berechnung weiterer "sekundärer" Ergebnisdaten zur numerischen und grafischen Anzeige und Auswertung der Ergebnisse.

Zur Nutzung seiner vollen Funktionalität benötigt SIRIUS noch einige Verzeichnisse, zur Sicherung der Berechnungsergebnisse und zur Kommunikation mit dem Grafikprogramm GNUPLOT und mit einem einfachen Filmschnittprogramm für die Erzeugung von Animationen. Als Filmschnittprogramm benutzte der Autor bisher das Programm Movie-Maker von Microsoft. In einigen weiteren Verzeichnissen sind diverse Hilfsdateien abgelegt, insbesondere zur Steuerung des Grafikprogramms GNOPLOT.

Falls auf den Quelltext zugegriffen werden soll, um das Programm an bestimmte Bedingungen anzupassen oder weiter zu entwickeln und anschließend zu kompilieren, wird ein FORTRAN-Compiler benötigt. Der Autor verwendet dafür den freien Compiler "GNU Fortran G77 for Win32" der aus dem Internet heruntergeladen und kostenlos genutzt werden kann.

Bild 3.01 zeigt die Grobstruktur des Programms SIRIUS und die erforderlichen Hilfsprogramme zur Erzeugung grafischer Darstellungen (graue Kästchen, schwarz umrandet). Die grün umrandeten gelben Kästchen zeigen die erforderlichen Verzeichnisse zur Datensicherung und zur Kommunikation der Programme untereinander. Die grünen Pfeile zeigen den Datenfluss. Die schwarzen Pfeile innerhalb des Programms SIRIUS zeigen die Möglichkeiten der Navigation innerhalb des Programms (Hauptdurchlaufrichtung: dicke Pfeile, Rücksprungmöglichkeiten: dünne Pfeile).

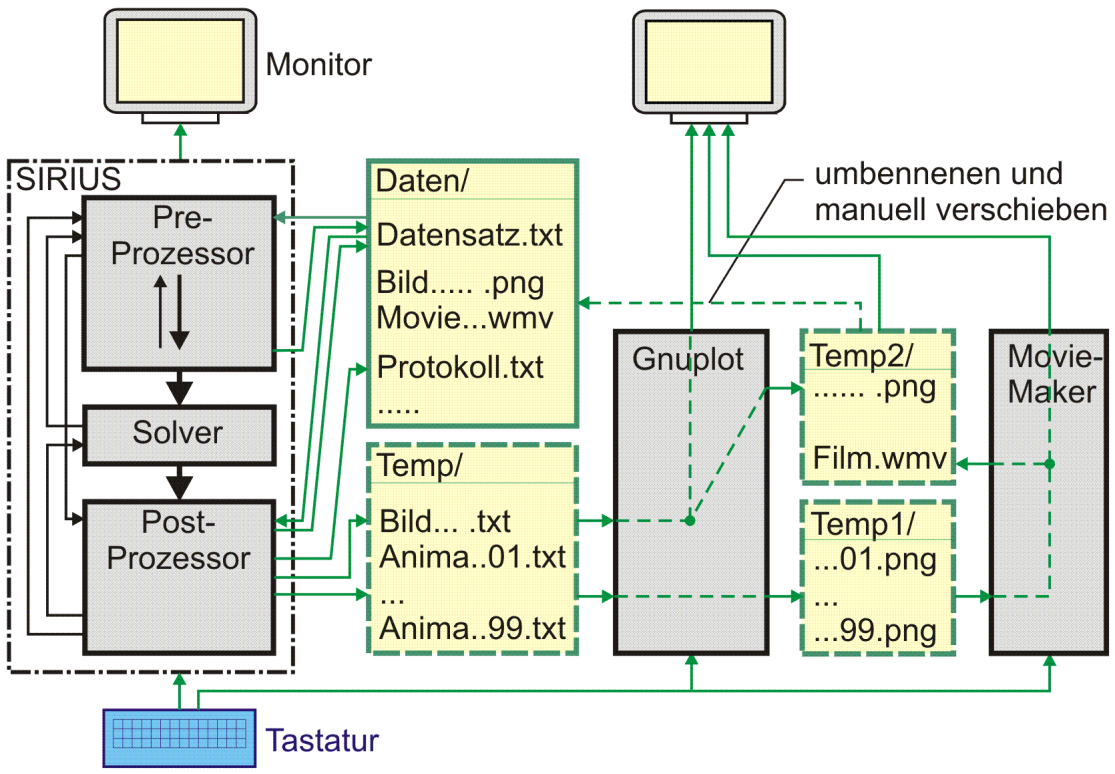


Bild 3.01: *Grobstruktur des Programm SIRIUS einschließlich erforderlicher Umgebung*

Das Programm GNUPLOT ist ein kostenlos nutzbares Programm und kann aus dem Internet geladen und installiert werden. Es muss mindestens die Version 4.6.3 geladen werden. Es ist nicht in das Programm SIRIUS integriert und muss als selbständiges Programm parallel dazu gestartet werden. Durch den automatisierten Zugriff beider Programme auf das Verzeichnis "Temp" ist eine zügige Datenübertragung gegeben, so dass man im Rahmen der Ergebnisauswertung mit beiden Programmen parallel arbeiten kann. Mit dem Filmschnittprogramm können dann aus einer Serie von Bildern, die GNUPLOT als PNG-Dateien im Verzeichnis Temp1 ablegt, Animationen erzeugt werden. Das verwendete Programm Movie-Maker ist bis Version Windows XP bereits im Betriebssystem enthalten. Für spätere Versionen des Windows-Betriebssystems kann es bei Microsoft kostenlos heruntergeladen werden. Es sind sicher auch einige andere einfache Filmschnittprogramme geeignet, da diese nur eine Reihe von Pixel-Grafiken zu einer fortlaufenden Animation zusammenfügen müssen.

Neben den im Bild 3.01 gezeigten Verzeichnissen zur Kommunikation der Programme untereinander und der Datensicherung sind weitere Verzeichnisse erforderlich mit diversen Dateien. Bild 3.02 zeigt die erforderliche Verzeichnisstruktur für die umfassende Nutzung des Programms.

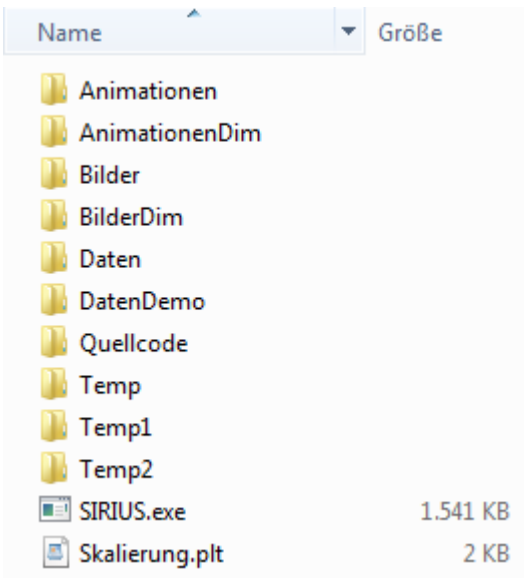


Bild 3.02: *Liste der Verzeichnisse für den Betrieb des Programms SIRIUS*

In den Verzeichnissen "**Animationen**", "**AnimationenDim**", "**Bilder**" und "**BilderDim**" sind Dateien mit der Erweiterung ".plt" abgelegt, die Befehlssätze zur Steuerung des Programms GNUPLOT enthalten. Jede dieser Dateien steuert die Erzeugung einer speziellen grafischen Ausgabe. Aktuell stehen Dateien für ca. 100 verschiedene Grafiken und Animationen zur Verfügung.

Im Verzeichnis "**Daten**" speichert das Programm die Dateien zur Sicherung der Eingabe- und Ergebnisdaten und liest sie zur weiteren Bearbeitung, Auswertung oder Ableitung neuer Berechnungsvarianten auch wieder aus diesem Verzeichnis ein.

Das Verzeichnis "**DatenDemo**" enthält alle Eingabe- und Ergebnisdatensätze, die in der Dokumentation zur Demonstration der Arbeitsweise des Programms beschrieben werden. Diese Datensätze können dazu genutzt werden, die beschriebenen Beispiele selbst nachzurechnen bzw. die Auswertung der Ergebnisse zu üben. Um ein Demonstrationsbeispiel zu testen, ist die jeweiligen Datei in das Verzeichnis "Daten" zu kopieren.

Das Verzeichnis "**Quellcode**" enthält alle Dateien des Quellcodes des FORTRAN-Programms SIRIUS und die Batchdatei "1-Start-CompilerG77.bat" zum Aufruf des FORTRAN-Compilers. Jede Routine ist in einer eigenen Datei gleichen Namens abgelegt. Es sind

außerdem einige Dateien enthalten, die aktuelle nicht im kompilierten Programm enthalten sind. Sie sind im Verzeichnis dadurch zu erkennen, dass sie die Erweiterung ".f_" haben und deshalb vom Compiler ignoriert werden. Siehe dazu auch Abschnitte 3.4.5.4, 4.9.3 und das Routinenverzeichnis 5.4.

Die Verzeichnisse "**Temp**", "**Temp1**" und "**Temp2**" dienen als Zwischenspeicher für die Kommunikation der drei beteiligten Programme untereinander, wie bereits in Bild 3.01 dargestellt.

Die Datei "**SIRIUS.exe**" enthält das komplette kompilierte ausführbare Programm SIRIUS.

Die Datei "**Skalierung.plt**" enthält die Skalierungsparameter für die Erzeugung von grafischen Darstellungen mit GNUPLOT. Siehe dazu Abschnitt 4.7.1.

3.2.2 Prinzipielle Abläufe der Hauptrechnung

In diesem Abschnitt soll ein Überblick über die prinzipiellen Berechnungsabläufe der Hauptrechnung im Solver gegeben werden. Die gesamte Variantenvielfalt des Programms lässt sich mit 4 Berechnungsabläufen abdecken, die sich im Wesentlichen durch die Anzahl der zu durchlaufenden Iterationsschleifen unterscheiden.

3.2.2.1 Druckverlauf1: klassische Reynoldssche Differentialgleichung, vorgegebene Wellenverlagerung

Die klassische Reynoldssche Differentialgleichung ist eine lineare partielle Differentialgleichung 2.Ordnung. Die näherungsweise Berechnung der Druckverteilung im Schmierspalt lässt sich deshalb mit einem Differenzenverfahren realisieren. Das führt auf ein lineares Gleichungssystem, für das es verschiedene numerische Lösungsverfahren gibt. Die Routine "FilmDruck1" realisiert im Programm SIRIUS diese Aufgabe.

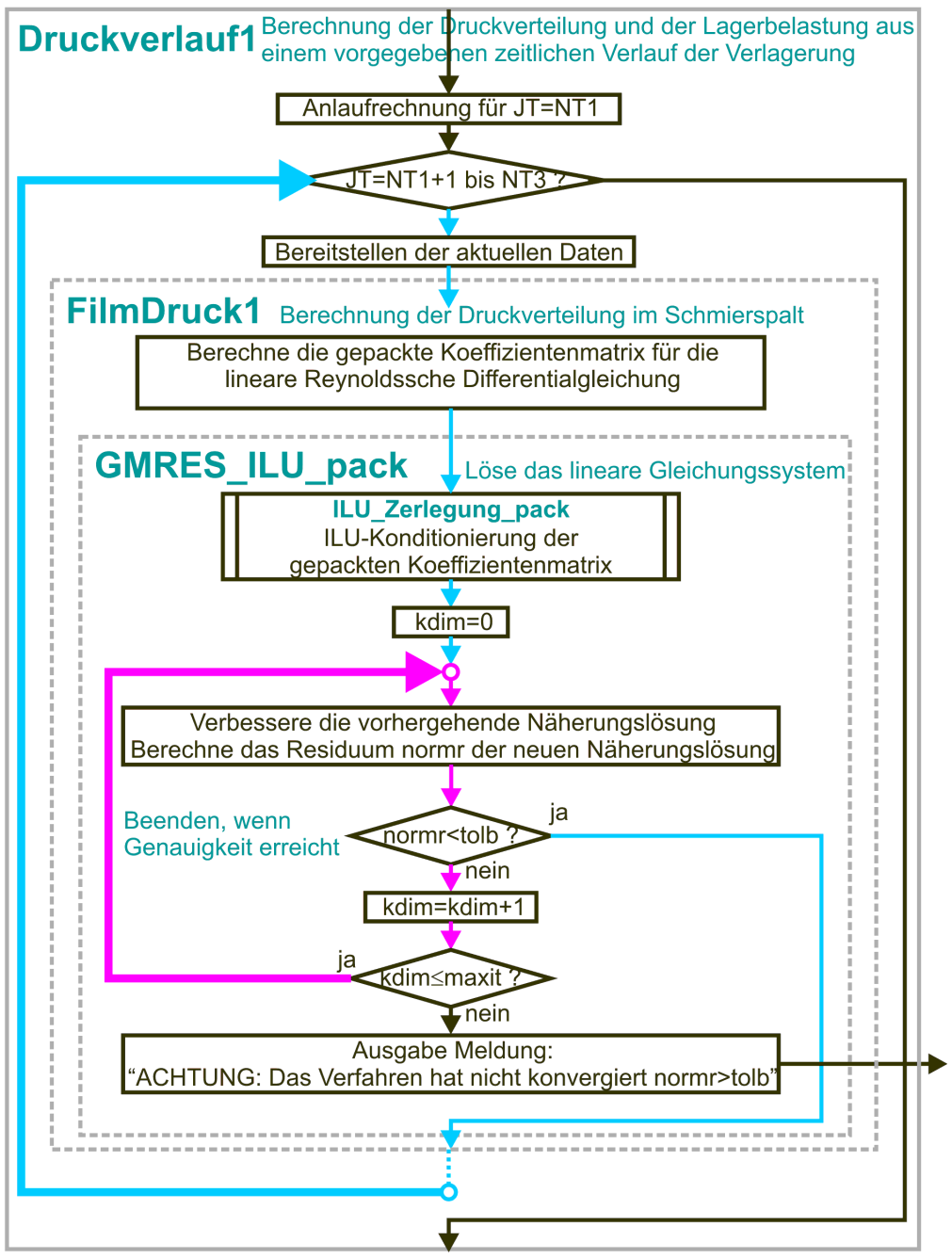


Bild 3.03: *Iterationsschleifen innerhalb der Routine "Druckverlauf1"*

Bild 3.03 zeigt ein Flussdiagramm mit einem stark vereinfacht dargestellten Berechnungsablauf der Routine "Druckverlauf1", in der die Routine "FilmDruck1" aufgerufen wird. In dieser wiederum ist die Routine "GMRES_ILU_pack" enthalten. Diese Routine enthält das Lösungsverfahren für das lineare Gleichungssystem. Die Rechenzeit für die Lösung des Gleichungssystems ist entscheidend für die Arbeitsgeschwindigkeit des gesamten Programms. Im Programm SIRIUS wird das **GMRES-Verfahren** (für **Generalized minimal residual method**) [11], [32] in Kombination mit einer Vorkonditionierung der Koeffizientenmatrix durch eine **ILU-Zerlegung** [10], [35] verwendet. Das GMRES-Verfahren arbeitet iterativ. Damit ist die innerste Iterationsschleife der Berechnung gegeben (magenta-farbene Schleife im Bild 3.03). Man könnte auch auf diese innerste Iterationsschleife verzichten, indem man ein direktes Verfahren verwendet, wie z.B. das Gaußsche Eliminationsverfahren, welches früher im Programm verwendet wurde. Bei einer Anzahl von ca. 10 000 Gleichungen, was für unsere Lagerberechnungen meist ausreichend ist, arbeitet das GMRES-Verfahren in Kombination mit der ILU-Zerlegung aber um den Faktor 1000 schneller. Hinzu kommt, dass hier mit einer gepackten Koeffizientenmatrix gearbeitet werden kann und so Speicherplatz gespart wird.

Sind die Betriebsbedingungen instationär, müssen die Druckverteilungen im Schmierspalt und die daraus resultierenden Lagerbelastungen über mehrere aufeinanderfolgende Zeitpunkte berechnet werden. Damit ist eine weitere äußere Schleife im Berechnungsablauf erforderlich (blaue Schleife im Bild 3.02). Im stationären Fall muss die äußere (blaue) Schleife nur einmal durchlaufen werden. Dieser Berechnungsablauf ist der einfachste und schnellste Ablauf der Hauptrechnung im Programm SIRIUS.

3.2.2.2 Druckverlauf2: erweiterte Reynoldssche Differentialgleichung, vorgegebene Wellenverlagerung

Die Berechnung der Druckverteilung im Schmierspalt und der Lagerbelastung aus einer vorgegebenen Wellenverlagerung, mit Hilfe der **erweiterten** Reynoldsschen Differentialgleichung, erfordert eine zusätzliche Iterationsschleife. Im Gegensatz zur klassischen

Reynoldsschen Gleichung ist die erweiterte Reynoldssche Gleichung nicht mehr linear. Durch eine Linearisierung der Gleichung, mit Hilfe einer Anfangsnäherung, die über mehrere Berechnungszyklen iterativ verbessert wird, lässt sich diese Gleichung ebenfalls mit einem modifizierten Differenzenverfahren numerisch lösen. Damit kommt eine weitere Iterationsschleife in den Berechnungsablauf.

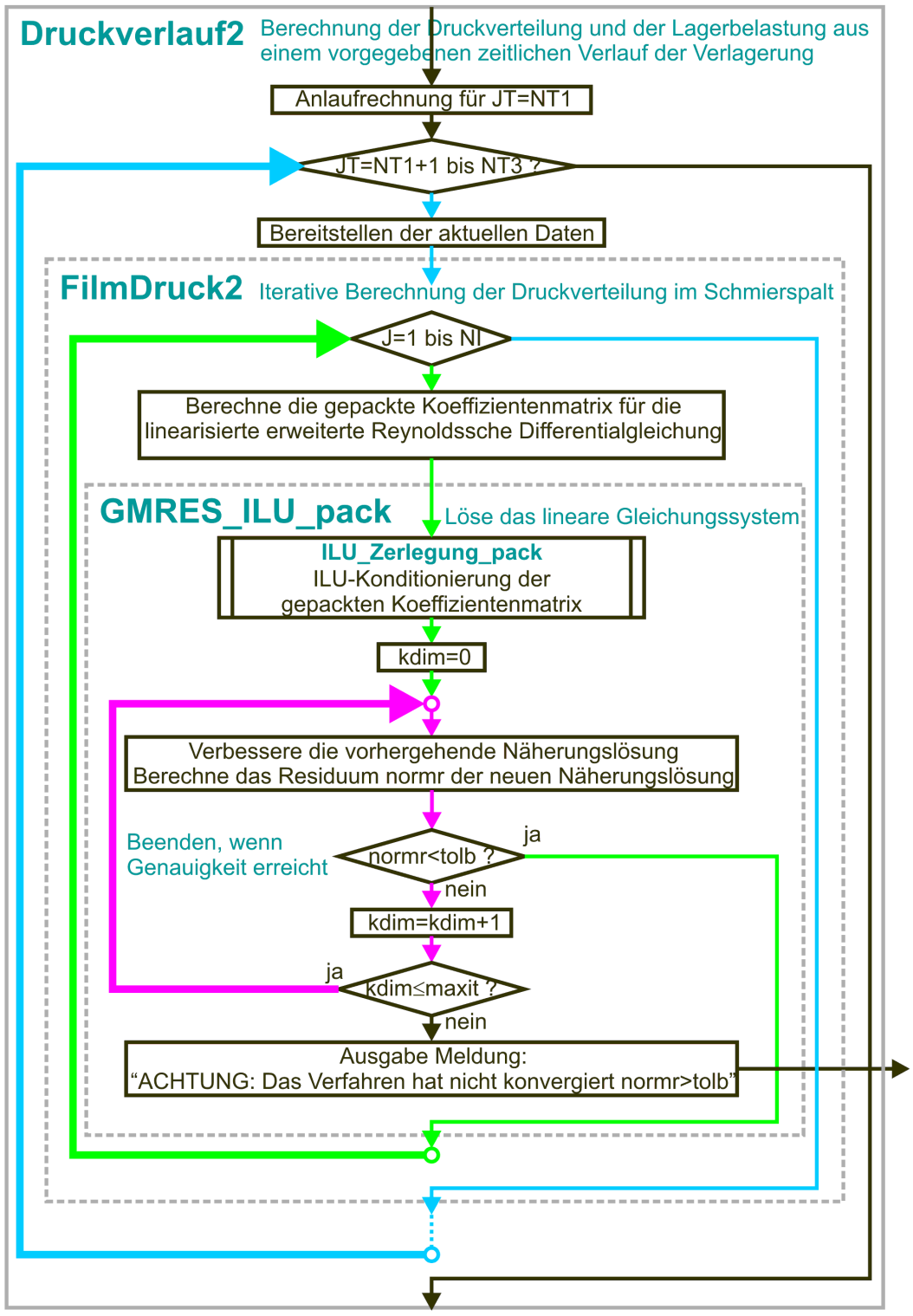


Bild 3.04: *Iterationsschleifen innerhalb der Routine "Druckverlauf2"*

Bild 3.04 zeigt ein Flussbild mit einem stark vereinfacht dargestellten Berechnungsablauf der Routine "Druckverlauf2" für die Berechnungen mit der erweiterten Reynoldsschen Gleichung. Die grüne Schleife ist die durch die Nichtlinearität der Gleichung zusätzlich hinzugekommene Iteration. Neben der Nichtlinearität kommt außerdem hinzu, dass die Druckverteilung im Schmierspalt nicht nur von der aktuellen Spaltgeometrie abhängt, sondern auch von der aktuellen Schmiermittelverteilung im nicht vollständig gefüllten Schmierspalt. Diese Verteilung hängt wiederum von den zeitlich vorausgehenden Verhältnissen im Schmierspalt ab. Deshalb ist hier immer eine Berechnung über mehrere Zeitschritte angesagt (blaue Schleife), selbst wenn nur die Druckverteilung eines stationär belasteten Lagers berechnet werden soll. Diese Berechnung über die Zeit stellt für den stationären Fall eine Anlaufrechnung dar, bei der sich die Lösung iterativ der stationären Endlösung nähert.

Auch für die Berechnung instationärer Lagerbelastungen, die meist als geschlossene Zyklen über eine oder mehrere Umdrehungen gegeben sind, ist zunächst eine Anlaufrechnung erforderlich, bevor der eigentliche in sich geschlossene Zyklus berechnet werden kann. Damit ist der äußere Berechnungszyklus (blaue Schleife) sowohl Iterationszyklus, als auch Simulation des eigentlichen physikalisch-technischen Verhaltens des Lagers. Bei kleinen Zeitschrittweiten sind die berechneten Druckverteilungen des vorhergehenden Zeitpunktes bzw. deren Extrapolation gute erste Näherungen für die iterative Berechnung des aktuellen Zeitpunktes. Das hat zur Folge, dass die inneren Iterationszyklen mit wenigen Durchläufen auskommen, so dass auch instationäre Betriebsbedingungen recht schnell berechnet werden können. Die Berechnung von beispielsweise 100 aufeinanderfolgenden Zeitpunkten eines instationären Lastfalles benötigt in der Regel wesentlich weniger Zeit als die Berechnung von 10 einzelnen stationären Lastfällen.

3.2.2.3 Verlagerungsbahn1: klassische Reynoldssche Differentialgleichung, vorgegebene Lagerbelastung

Das übliche ingenieurtechnische Problem der Lagerberechnung besteht darin, dass eine Lagerbelastung vorgegeben ist und daraus die Wellenverlagerung und die minimale Schmierspalthöhe zu ermitteln ist. Damit ist die Richtung von Ursache und Wirkung entgegengesetzt zu ihrer Darstellung in der Reynoldsschen Differentialgleichung. Ausgehend von dem prinzipiellen Ablauf der Berechnung in Bild 3.03 für die klassische Reynoldssche Differentialgleichung ist damit ein weiterer Iterationszyklus erforderlich.

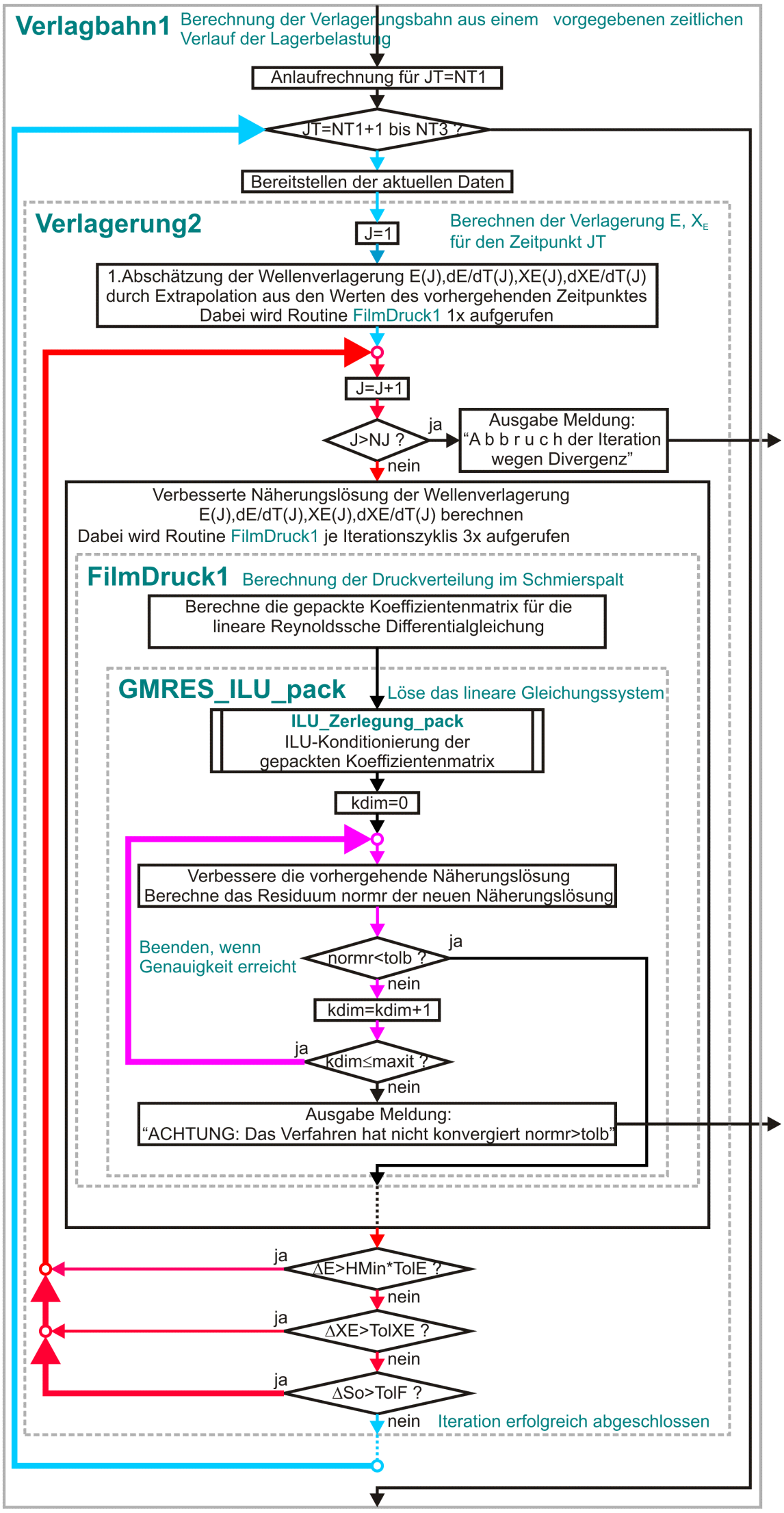


Bild 3.05: *Iterationsschleifen innerhalb der Routine "Verlagbahn1"*

Bild 3.05 zeigt ein Flussbild mit einem stark vereinfacht dargestellten Berechnungsablauf der Routine "Verlagbahn1" für die Berechnung der Wellenverlagerung aus vorgegebenen Lagerbelastungen mit der klassischen Reynoldsschen Differentialgleichung. Die rote Schleife ist hier der neu hinzugekommene Iterationszyklus. Alternativ zur Routine "Verlagerung2" kann auch die Routine "Verlagerung1" zur Anwendung kommen. Ihre innere Struktur ist bezogen auf die Iterationsschleifen identisch zur Routine "Verlagerung2".

3.2.2.4 Verlagerungsbahn2: erweiterte Reynoldssche Differentialgleichung, vorgegebene Lagerbelastung

Für die Anwendung der erweiterten Reynoldsschen Differentialgleichung auf die Berechnung der Wellenverlagerung aus vorgegebenen Lagerbelastungen ergibt sich nun, in Kombination der Abläufe der Bilder 3.04 und 3.05, ein Berechnungsablauf mit 4 ineinander geschachtelten Iterationsschleifen gemäß Bild 3.06.

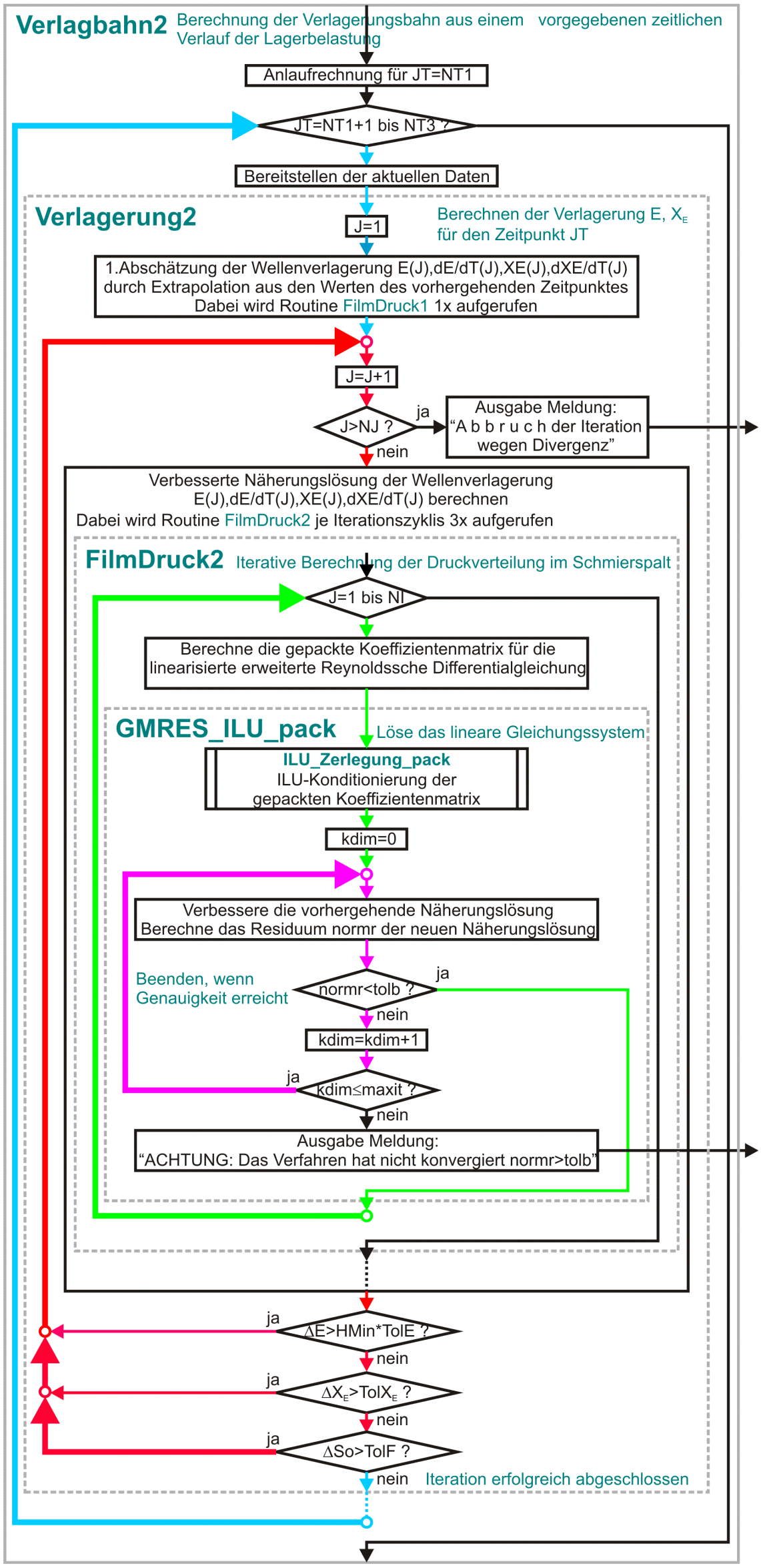


Bild 3.06: *Iterationsschleifen innerhalb der Routine "Verlagbahn1"*

Dieser Berechnungsablauf ist der aufwendigste. Er beschreibt aber die Verhältnisse im Lager am realistischsten und wird deshalb am häufigsten angewendet. Auch hier gilt für instationäre Lastfälle, wie bereits oben erwähnt, dass die Lösungen der vorher liegenden Zeitpunkte gute Näherungen für die aktuellen Zeitpunkte sind und so die 3 inneren Iterationszyklen nicht oft durchlaufen werden müssen, so dass sich in der Regel immer noch kurze Rechenzeiten ergeben. Alternativ zur Routine "Verlagerung2" kann auch die Routine "Verlagerung1" zur Anwendung kommen. Ihre innere Struktur ist bezogen auf die Iterationsschleifen identisch zur Routine "Verlagerung2".

3.2.3 Eine etwas detaillierte Übersicht über die Programmstruktur

Neben vielen recht komplexen Teilaufgaben, die innerhalb der Hauptrechnung ablaufen, befassen sich etwa 2/3 der Routinen mit der Dateneingabe im PreProzessor und der Ergebnisdarstellung und -sicherung im Postprozessor. Das konkrete Zusammenspiel dieser Routinen ist im Detail nur aus dem Quelltext und dem Routinenverzeichnis (siehe Anhang 5.4) herauszulesen, was unübersichtlich und

beschwerlich ist. Um das Verständnis der Arbeitsweise des Programms auf anschauliche Art weiter zu vertiefen, wird die Struktur des Programms in den nachfolgenden Abschnitten durch wesentlich detaillierte Flussbilder dargestellt. Auch diese Darstellungen stellen noch nicht jedes Detail des Programmablaufs dar, um die Übersichtlichkeit zu bewahren. Ein Großteil der Unterroutinen wird auch hier nur als ein Kästchen dargestellt, in dem der Name der Routine und ihre Aufgabe angegeben wird.

3.2.3.1 Die Hauptroutine "SIRIUS"

Die Hauptroutine "SIRIUS" ist noch sehr übersichtlich, da sie nur die Titelzeilen des Programms auf dem Bildschirm anzeigt und als Rahmen für den Aufruf der 3 Routinen "PreProzessor", "Solver" und "PostProzessor" fungiert.

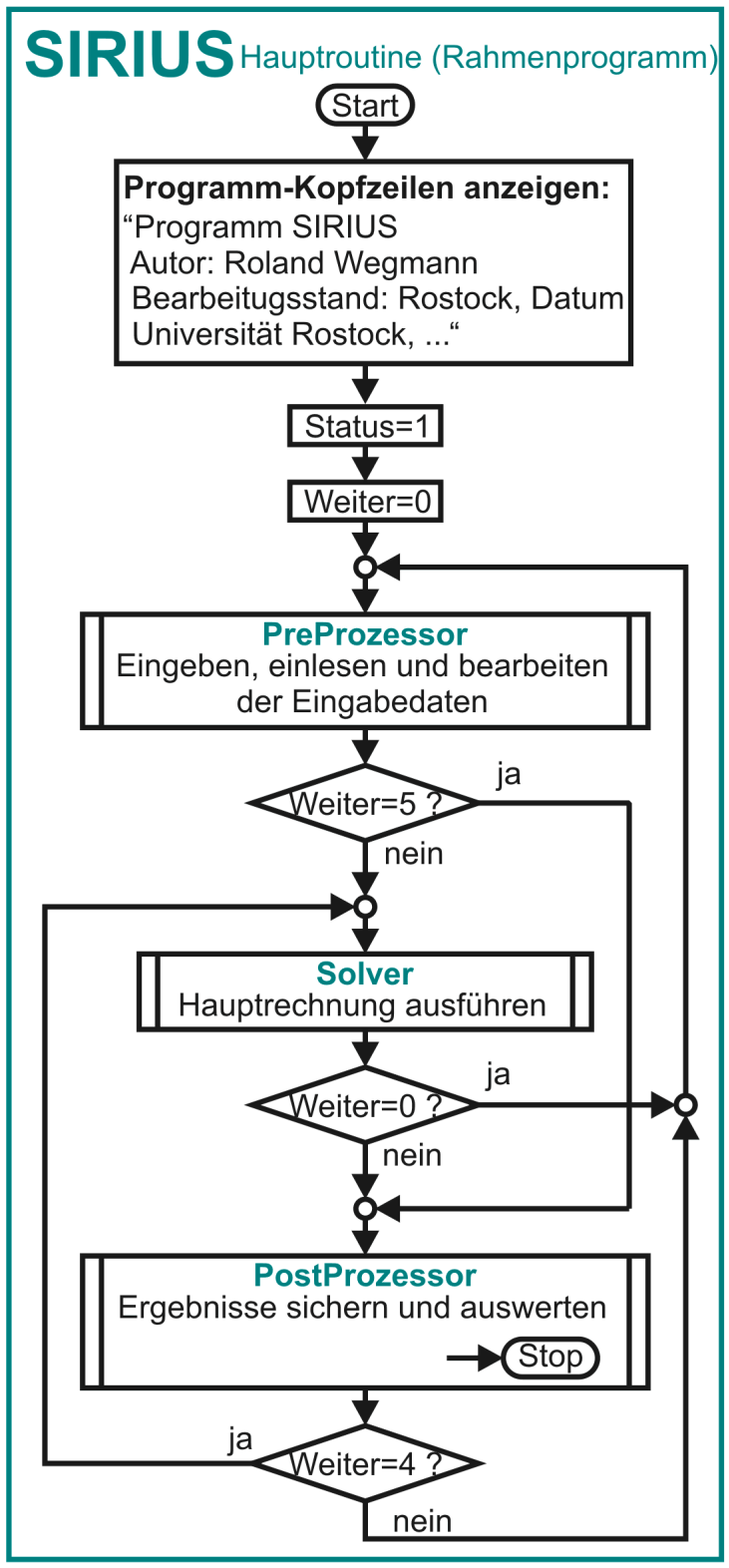


Bild 3.07: *Flussbild der Hauptroutine "SIRIUS"*

Bild 3.07 zeigt das Flussdiagramm der Hauptroutine. Die Routine operiert nur mit den zwei Steuerparametern "Status" und "Weiter". Die Variable "Status" wird hier auf den Startwert 1 gesetzt und wird in den untergeordneten Routinen weiter manipuliert und ausgewertet. Sie gibt Auskunft über Zustände des Programms und löst die Anzeige von Meldungen aus. Sie kann folgende Werte annahmen:

Status = 1 Das Programm wurde neu gestartet, die Standard-Eingabeparameter müssen initialisiert werden.
= 2 Die Eingabedaten sind in Bearbeitung. Die Bearbeitung ist noch nicht bis zum Ende durchlaufen. Die Eingabedaten können noch inkonsistent sein.
= 3 Es wurde ein Datensatz aus einer Datei eingelesen. Es kann direkt zur Berechnung übergegangen werden. Es ist aber nicht sicher, dass die Daten konsistent sind, weil sie evtl. außerhalb des Programms manipuliert wurden.
= 4 Bearbeitung der Eingabedaten wurde vollständig durchlaufen. Es liegt ein konsistenter Eingabedatensatz vor, mit dem die Berechnung durchgeführt werden kann.
= 5 Fortsetzung der Bearbeitung nach rechentechnischen Problemen.
= 6 Die Berechnung wurde abgebrochen. Das GMRES-Verfahren konvergiert nicht.
= 7 Die Berechnung der Verlagerung wurde abgebrochen, weil die Iteration über NJ Zyklen nicht konvergiert.
= 8 Die Berechnung wurde abgebrochen, weil eine Spalthöhe $H \leq 0$ auftrat.
= 9 Berechnung wurde nicht begonnen, weil die zulässige Anzahl $N_{GleiMax}$ von Gleichungen überschritten wurde oder der Gleichungslöser SIMQ hat einen Fehler gemeldet. (Kann nur auftreten, wenn mit ungepackter Matrix gearbeitet wird.)

Die Steuervariable "Weiter" wird durch Eingaben des Anwenders zur Art der gewünschten Fortsetzung des Programmablaufs gesetzt und dementsprechend vom Programm ausgewertet. Sie kann folgende Werte annehmen:

Weiter = 0 Gehe zum PreProzessor, zum Anfang der Eingabe
= 1 Gehe zurück zum vorhergehenden Menü
= 2 Gehe weiter zum nächsten Menü
= 3 Wechsel in die dimensionslose bzw. dimensionsbehaftete Darstellung
= 4 Gehe zum Solver
= 5 Gehe zum PostProzessor

3.2.3.2 PreProzessor

Die Routine "PreProzessor" umfasst die gesamte Dateneingabe und die weitere Datenaufbereitung für die Hauptrechnung.

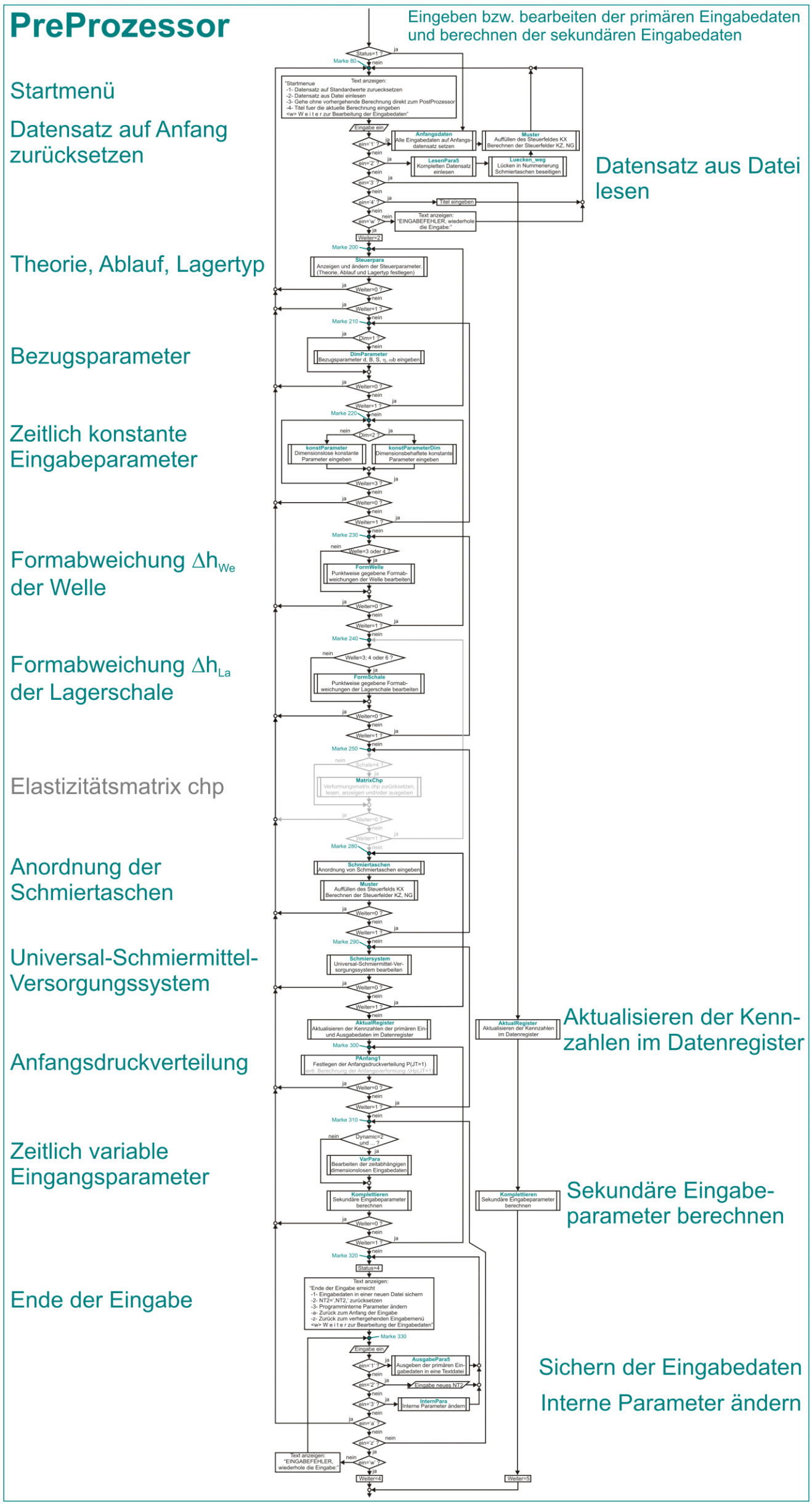


Bild 3.08: *Flussdiagramm der Routine "PreProzessor"*

HINWEIS: *Das Bild ist wegen seiner Länge hier zunächst stark verkleinert. Wer mehr Details erkennen möchte, kann sich durch klicken auf das Bild eine vergrößerte Version anzeigen lassen.*

Bild 3.08 zeigt das Flussdiagramm der Routine. Der Ablauf spiegelt sich recht gut in der Abfolge der Eingabemenüs wieder, wie sie dem Anwender nacheinander am Bildschirm erscheinen. Siehe dazu auch Bild 4.009. Er bekommt aber nur die Menüs zu Gesicht, die für die aktuelle Variante relevant sind. Die meisten der hier dargestellten Unterroutinen repräsentieren ein Menü, das eine Gruppe von Eingabeparametern zur Auswahl und Bearbeitung zusammenfasst. Jede dieser Routinen ruft dann weitere Unterroutinen auf zur konkreten Bearbeitung einzelner Parameter. So verbergen sich hinter den schlichten Kästchen dieses Flussdiagramms weitere teils recht komplexe Strukturen des Programms.

Der Anwender braucht sich während der Eingabe nur mit den primären Eingabedaten zu befassen. Von ihm unbemerkt, werden vom Programm weitere sekundäre Eingabedaten berechnet bzw. ermittelt. Das erfolgt im Wesentlichen durch die Routinen "Muster", "AktualRegister" und "Komplettieren".

Die Routine "Muster" füllt das Feld KX, in dem die vom Anwender eingegebene Daten der Schmiertaschenverteilung bereits enthalten sind, mit weiteren sekundären Steuerdaten auf, die das Programm bei der Anwendung des Differenzenverfahrens zur Auswahl der richtigen Differenzengleichung benötigt. Diese sekundären Steuerdaten werden bei der Anzeige des Steuerfeldes KX jedoch unterdrückt, da der Anwender diese Information nicht wissen muss und dadurch die Übersichtlichkeit des Eingabemenüs besser ist. Außerdem füllt die Routine "Muster" die Felder KZ und NG mit sekundären Steuerdaten. Diese Steuerdaten habe eine analoge Aufgabe, wie die sekundären Steuerdaten in KX. Die Felder KZ und NG bleiben dem Anwender vollständig verborgen.

Die Routine "AktualRegister" ermittelt einen umfangreichen Satz von sekundären Steuerdaten, die für die Steuerung der Eingabe zeitvariabler Eingabedaten und für die Sicherung der primären Eingabe- und Ergebnisdaten in einer externen Datendatei benötigt werden.

Die Routine "Komplettieren" berechnet alle restlichen Eingabedaten, die das Programm für die Hauptrechnung benötigt und durch die eingegebenen primären Eingabedaten bereits feststehen.

Neben dem mittigen Hauptweg durch die Routine "PreProzessor" gibt es rechts im Bild 3.08 einen Nebenweg, der aus dem Startmenü direkt an das Ende der Routine führt. Falls im Startmenü aus einer externen Datei ein kompletter Datensatz von primären Eingabe- und Ergebnisdaten einer früheren Berechnung geladen wurde, kann man auf diesem Weg unter Umgehung der Eingaberoutinen direkt zum PostProzessor gelangen, um diese Daten auszuwerten. Auf diesem Weg werden lediglich die 3 oben erläuterten Routinen "Muster", "AktualRegister" und "Komplettieren" aufgerufen zur Ergänzung der zugehörigen sekundären Eingabedaten", da diese bei der Datensicherung nicht gespeichert wurden. Die sekundären Ergebnisdaten, die ebenfalls nicht abgespeichert wurden, brauchen hier nicht rekonstruiert werden. Der PostProzessor erledigt das auf entsprechende Anfrage selbst.

Dieser Bypass im Programm ist ein Kompromiss zu Gunsten der Flexibilität und zu Lasten der Zuverlässigkeit des Programms. Es könnte sein, dass die eingelesenen Daten extern unsachgemäß manipuliert wurden. Durch die Umgehung der Konsistenzprüfung in den Eingaberoutinen und einen möglichen Rücksprung aus dem PostProzessor in den Solver könnten Berechnungen erfolgen, die zu unerwarteten Ergebnissen oder auch zum Absturz des Programms führen. Der Nutzer muss sich dieser Gefahr bewusst sein. Das Programm gibt beim Beschreiten dieses Pfades im PostProzessor eine Warnung aus.

3.2.3.3 Solver

Die Routine "Solver" umfasst die gesamte Hauptrechnung. Hier werden die primären Ergebnisdaten berechnet. Das ist das Feld der Druckverteilungen $P(N_Z,N_X,N_{T2})$ im Schmierspalt über das Zeitintervall von N_{T2} Zeitpunkten. Hinzu kommen die Drücke und Ölströme im peripheren Schmiermittel-Versorgungssystem, falls es vorhanden ist, und je nach Variante die resultierenden Lagerbelastungen $So(N_{T2})$, $X_{So}(N_{T2})$ bzw. die Verlagerungsbahn $E_1(N_{T2}),E_2(N_{T2})$ bzw. $E(N_{T2}),X_E(N_{T2})$ und der Verlauf der minimalen Schmierspalthöhe $H_{Min}(N_{T2})$ über die N_{T2} Zeitpunkte.

Bild 3.09 zeigt das Flussdiagramm der Routine.

HINWEIS: *Das Bild ist wegen seiner Größe hier zunächst stark verkleinert. Wer mehr Details erkennen möchte kann durch klicken auf das Bild eine vergrößerte Version anzeigen lassen.*

Zu Beginn der Routine kann der Nutzer festlegen, ob eine komplette Berechnung über alle Zeitpunkte erfolgen soll oder nur über einige ausgewählte Zeitschritte. Dann muss er das Ende abwarten und erhält nur die kurze Informationen, welche Zeitpunkte das Programm bereits erledigt hat. Wenn eine Verlagerungsbahn berechnet wird, kommen noch einige Informationen zum Konvergenzverhalten der Berechnung dazu. Von dem ansonsten recht komplexen Berechnungsablauf, wie er aus dem Flussdiagramm auch nur teilweise zu erkennen ist, sieht der Anwender nichts.

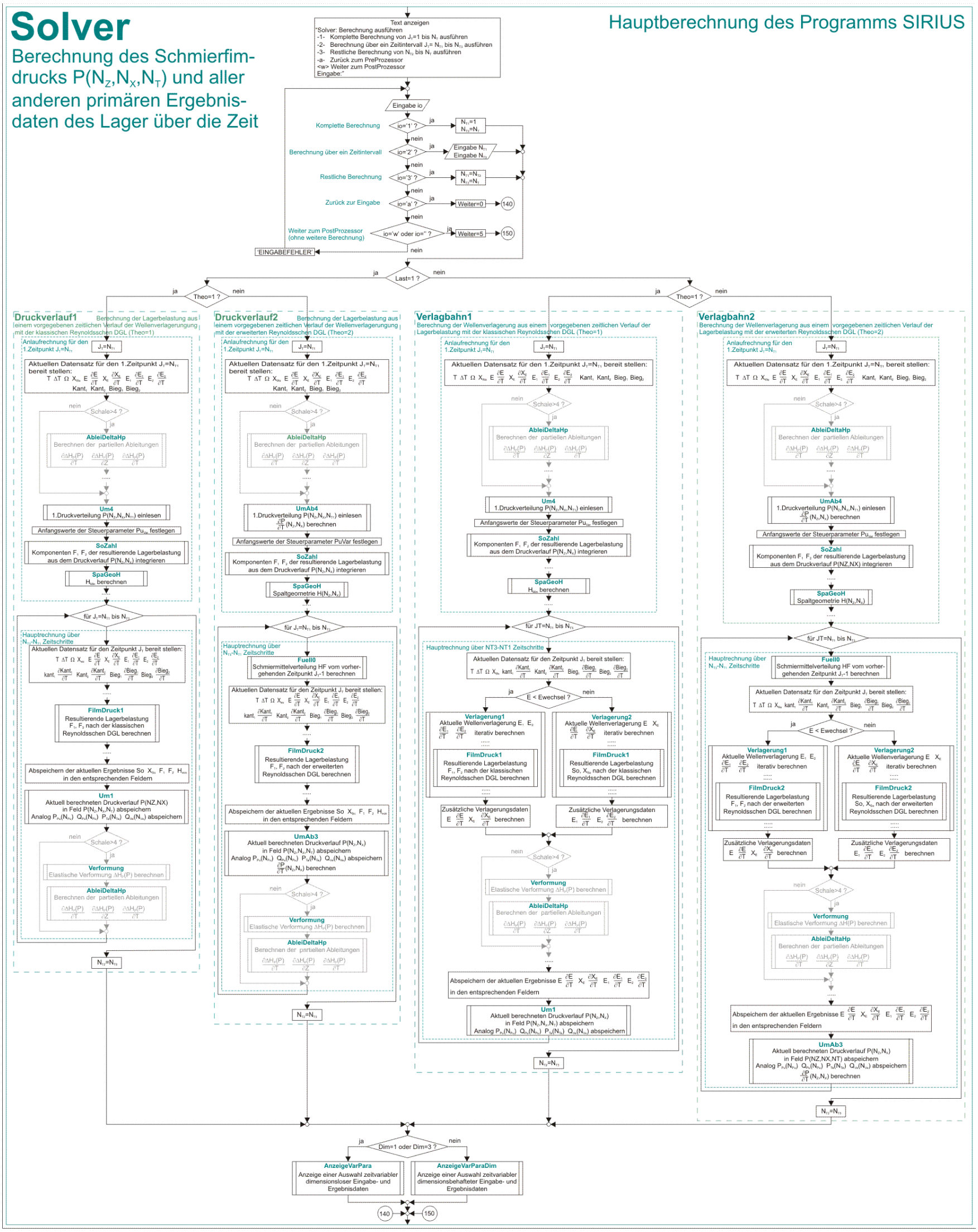


Bild 3.09: *Flussdiagramm der Routine "Solver"*

Hier im Flussdiagramm sind wieder die 4 möglichen Berechnungsvarianten als 4 parallele Berechnungsabläufe zu erkennen, die im Abschnitt 3.2.2 bereits erläutert wurden. Entsprechend der gewählten Variante wird jeweils ein Strang durchlaufen. In den beiden rechten Strängen (Routinen "Verlagbahn1" und "Verlagbahn2") erfolgt eine weitere Aufteilung der Stränge in die Routinen "Verlagerung1" und "Verlagerung2". Hier werden die jeweils aktuellen Wellenverlagerungen iterativ berechnet. Die beiden Stränge unterscheiden sich lediglich dadurch, dasss im linken Zweig "Verlagerung1" die Iteration in kartesischen Koordinaten E_1, E_2 erfolgt, während im rechten Zweig "Verlagerung2" die Iteration in polaren Koordinaten E, X_E erfolgt. Der linke Zweig wird bei kleiner Exzentrizität angewendet und der rechte Strang bei großer Exzentrizität. Weitere Erläuterungen dazu siehe Abschnitt 3.4.7. Die grau dargestellten Abschnitte des Flussdiagramms sind zukünftig erforderlich, wenn die elastische Verformung des Lagers berücksichtigt werden soll.

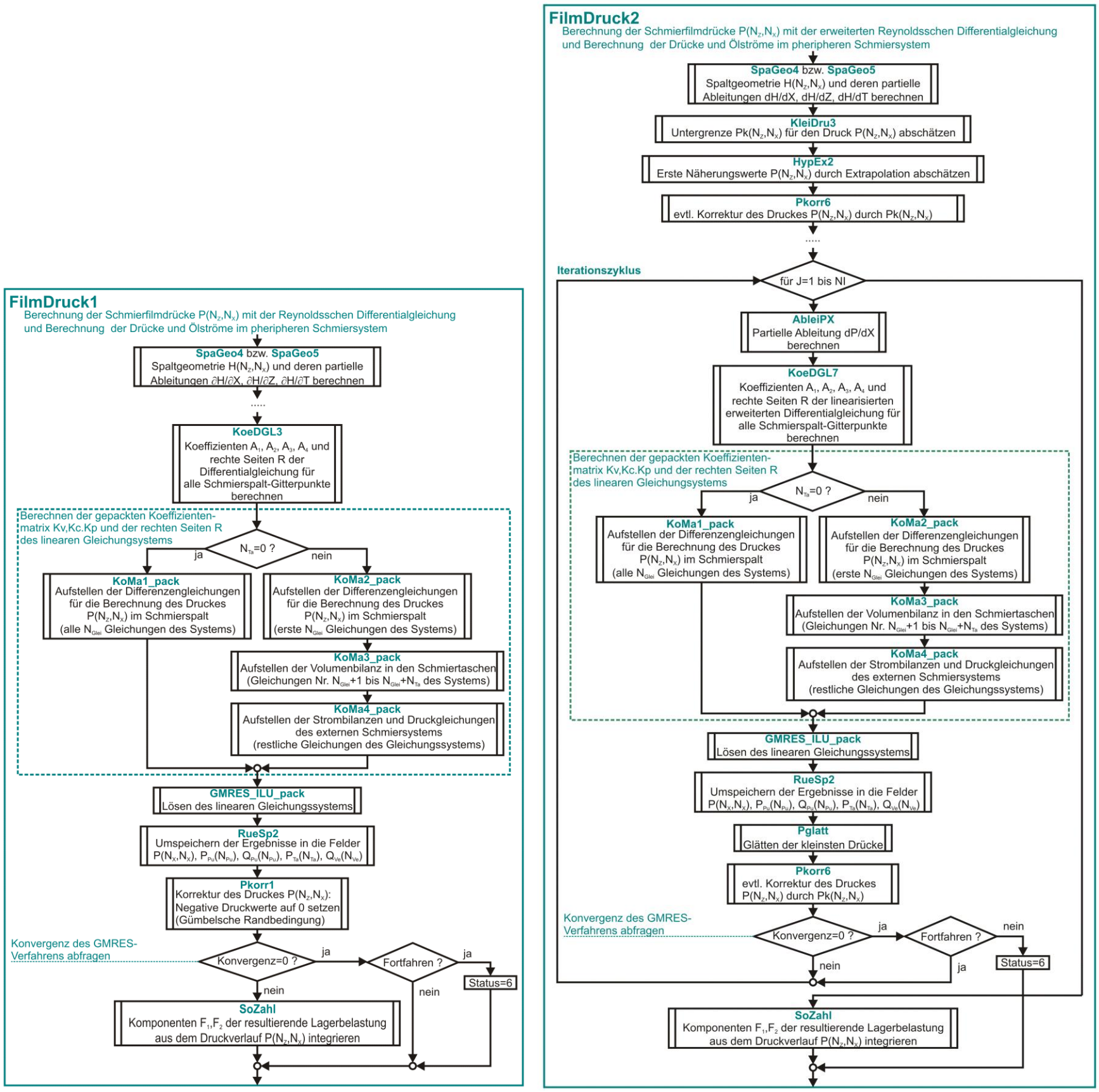


Bild 3.10: *Flussdiagramme der Routinen "FilmDruck1" und "FilmDruck2"*

Zwei wichtige Routinen, die im Bild 3.09 nur als kleine Kästchen mehrfach auftreten, sind "FilmDruck1" und "FilmDruck2". Diese enthalten den rechenintensiven Hauptteil, die Berechnung der Druckverteilung im Schmierfilm einschließlich der Drücke und Ölströme in der peripheren Schmiermittelversorgung. Die Routine "FilmDruck1" rechnet mit der klassischen Reynoldsschen Differentialgleichung und die Routine "FilmDruck2" mit der erweiterten Reynoldsschen Differentialgleichung.

3.2.3.4 PostProzessor

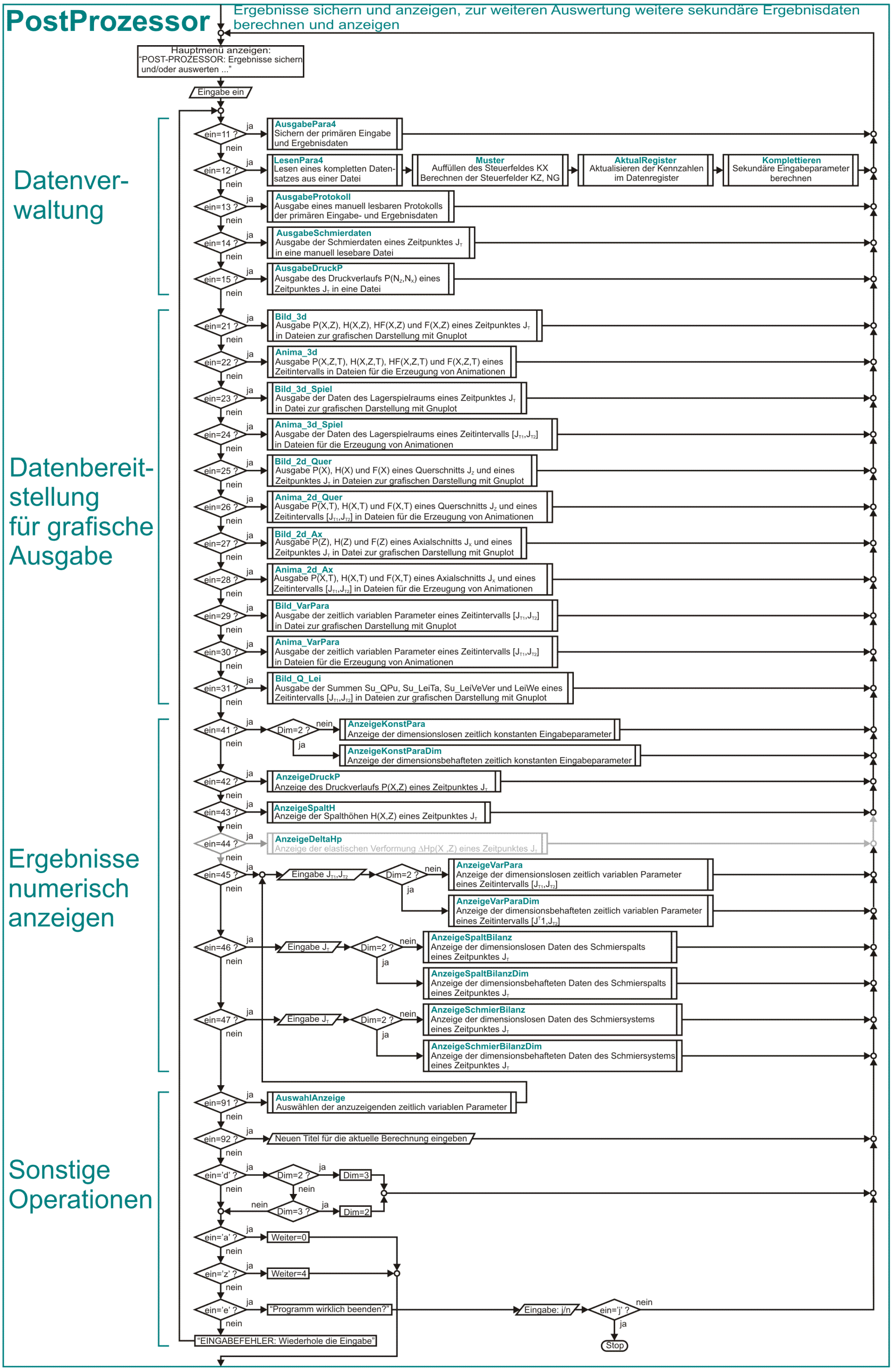


Bild 3.11: *Flussdiagramm der Routine "PostProzessor"*

HINWEIS: *Das Bild ist wegen seiner Länge hier zunächst stark verkleinert. Wer mehr Details erkennen möchte, kann sich in der HTML-Version durch klicken auf das Bild eine vergrößerte Version anzeigen lassen.*

Bild 3.11 zeigt das Flussdiagramm der Routine "PostProzessor". Hier sind alle Unterroutinen versammelt, die der Datensicherung sowie der grafischen und numerischen Darstellung der Ergebnisse dienen. Die Grundstruktur zeigt sich dem Anwender im Hauptmenü dieser Routine (siehe dazu auch Abschnitt 4.6). Hier kann man aus einer Reihe möglicher Operation eine auswählen und ausführen lassen. Nach Abschluss der Operation, die in der Regel noch durch einige Untermenüs führt, springt das Programm zurück in das Hauptmenü und ist bereit zur Ausführung weiterer Operationen.

Die Gesamtheit der möglichen Operationen umfasst 4 Blöcke:

1. Datenverwaltung
2. Bereitstellung von Daten für grafische Darstellungen mit GNUPLOT
4. Numerische Darstellung der Ergebnisse
9. Sonstige Operationen

Wenn bereits in Dateien gesicherte Berechnungsergebnisse existieren, kann die Routine "PostProzessor" auch selbständig operieren, ohne vorher den "PreProzessor" und den "Solver" zu durchlaufen. Dazu gibt es im Block der Datenverwaltung die Möglichkeit einzelne Datensätze einzulesen und anschließend auszuwerten. Jeder neu eingelesene Datensatz wird hier sofort nach dem Lesen durch die Routinen "Muster", "AktualRegister" und "Komplettieren" mit den notwendigen sekundären Eingabedaten vervollständigt, was sonst bereits im "PreProzessor" erfolgt ist.

3.2.4 Das Zusammenspiel der Routinen

Das Programm SIRIUS umfasst ca. 700 Seiten kommentierten Quelltext, der sich auf ca. 120 Routinen verteilt. Es wurde von Anfang an als Baukastensystem aufgebaut. Dadurch war eine kontinuierliche Weiterentwicklung möglich. Die Aufteilung der Routinen erfolgte hauptsächlich nach funktionellen Gesichtspunkten und zum Teil auch nach programmiertechnischen Erfordernissen.

Bild 3.12 gibt einen optischen Überblick über das Zusammenspiel der einzelnen Routinen. Jedes Kästchen symbolisiert genau eine Routine. Die Pfeile zwischen den Routinen symbolisieren den Zugriff der einzelnen Routinen auf die jeweiligen Subroutinen einer darunter liegenden Hierarchieebene. Die Pfeile zeigen von der aufrufenden Routine auf die aufgerufenen Routinen. Es hat sich inzwischen eine Tiefe der Strukturierung von maximal 7 Hierarchieebenen ergeben. Die grau hinterlegten Routinen gehören nicht zum Bestand des aktuellen Programms. Es sind zum Teil Routinen, die nur für Testzwecke vorgesehen sind und dazu gegebenen Falls erst aktiviert werden müssen. Außerdem sind es Routinen, die bereits eingebaut sind für die Berücksichtigung elastischer Verformungen der Lagerschale, die aber noch nicht genutzt werden können, weil das Berechnungsverfahren noch nicht ausgereift ist und weiterer Entwicklungsbedarf besteht.

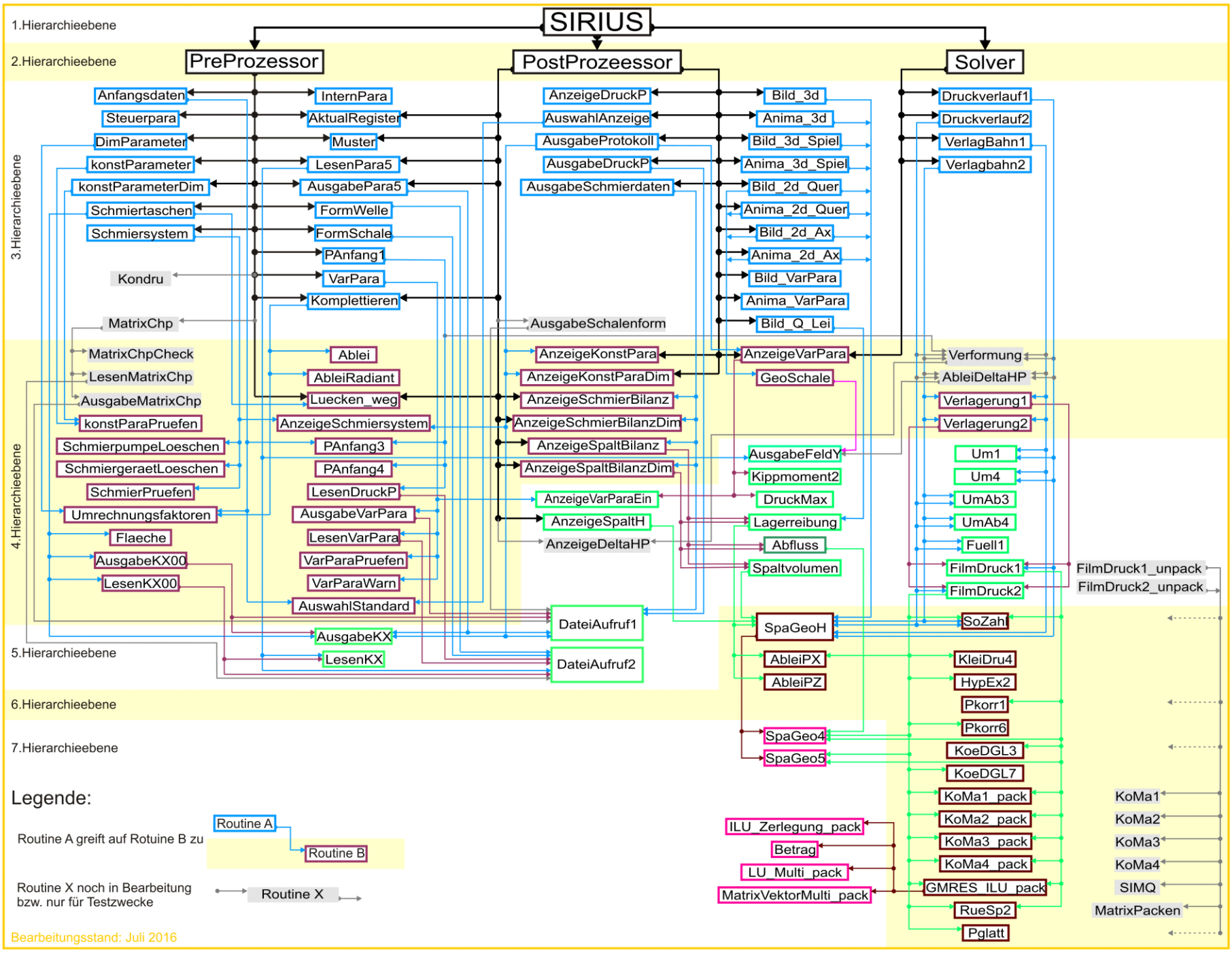


Bild 3.12: *Das Zusammenspiel der Routinen*

Das Bild 3.12 ist geeignet für einen optischen Überblick. Als zuverlässige Programmierhilfe im Rahmen einer Erweiterung oder Modifizierung des Programms ist es nicht geeignet, weil seine Aktualisierung sehr umständlich ist und deshalb eine ständige Aktualität nicht gewährleistet werden kann. Dazu besser geeignet ist das Verzeichnis der Routinen (siehe Anhang 5.4). Das Bild 3.12 wurde aufgrund der Angaben dieses Verzeichnisses erstellt. Das Routinenverzeichnis enthält noch weitere hilfreiche Informationen, die für eine fehlerfreie Bearbeitung des Gesamtprogramms wichtig sind. Solltest Du Dich entschließen am Quellcode des Programms zu arbeiten, ist eine laufende penible Aktualisierung dieses Verzeichnisse dringen angeraten. Z.B. muss die rechte Spalte des Verzeichnisses unbedingt auf dem aktuellen Stand sein, wenn die Feldgrößen im Programm verändert werden sollen, um evtl. mit noch feineren Gitterteilungen ΔX und ΔZ oder mehr Zeitschritten N_T arbeiten zu können. Siehe dazu Abschnitt 4.2.9.

3.3 Die Datenstruktur des Programms und ihre Verwaltung

Die Vielfalt der darstellbaren Lagervarianten erfordert eine große Anzahl von Eingabe- und Ergebnisparametern. Hinzukommen weitere Variablen zum Speichern von Zwischenergebnissen und Hilfswerten. Im Programm sind dafür mehr als 600 verschiedene Variablen definiert, von denen einige umfangreiche Felder mit vielen Werten repräsentieren. Von diesen Variablen können ca. 200 an der Programmoberfläche erscheinen. Um bei der Programmierung und der Anwendung des Programms nicht den Überblick zu verlieren, erfordert das eine systematische Datenstruktur, die nachfolgend beschrieben wird.

3.3.1 Ein einheitliches Bezeichnungssystem

Für die Parameter, die an der Programmoberfläche erscheinen, und die globalen Variablen, die in mehreren Routinen bearbeitet werden, wurde eine weitgehend einheitliche sprechende Nomenklatur erarbeitet: Jede Variable besteht aus einem **Basissymbol**, dass aus mehreren Zeichen bestehen kann. Beschreibt das Symbol einen dimensionslosen Parameter, beginnt es mit einem großen Buchstaben. Beschreibt es einen dimensionsbehafteten Parameter, beginnt es mit einem kleinen Buchstaben. Evtl. vorhandene weitere Zeichen des Basissymbols sind in der Regel kleine Buchstaben.

Jedes Basissymbol kann durch **Bezeichnungszusätze** ergänzt werden. Diese Zusätze können ebenfalls aus mehreren Zeichen bestehen, beginnend mit einem großen Buchstaben und evtl. gefolgt von weiteren kleinen Buchstaben. Es können auch Ziffern sein. Im Dokumentationstext werden die Bezeichnungszusätze als tiefer gestellte Zeichen dargestellt (Indexschreibweise). Im Quelltext und an der Programmoberfläche werden die Zusätze einfach an das Basissymbol angehängt, da die Programmiersprache FORTRAN die Indexschreibweise nicht zulässt. Ein Basissymbol kann auch durch mehrere Zusätze ergänzt werden. Indem die Bezeichnungszusätze mit einem großen Buchstaben beginnen, können in der einfachen Verkettung von Basissymbol und einem oder mehreren Zusätzen die Namensstruktur in der Regel leicht erkannt werden. Zur Verbesserung der Erkennbarkeit werden gelegentlich auch Unterstriche eingefügt.

Die Bezeichnungszusätze "..._0", "..._Ak" und "..._k" werden an eine größere Anzahl von Basisbezeichnungen angefügt. Zu ihrer Bedeutung siehe Tabelle 3.2. Da die Parameter mit diesen Bezeichnungszusätzen nur programmintern verwendet werden, sind sie nicht extra im Symbolverzeichnis aufgeführt.

Alle Parameter mit dem **Bezeichnungszusatz "_0"** sind im Common-Block "common/Datenregister/" abgelegt.

Alle Parameter mit dem **Bezeichnungszusatz "_k"** sind im Common-Block "common/konstVarParameter/" abgelegt.

Alle Parameter mit dem **Bezeichnungszusatz "_Ak"** sind in den Common-Blöcken "common/aktuelleVarParameter/" und "common/Druck/" abgelegt.

Echte Indizes zur Bezeichnung einer Komponente eines Datenfeldes werden in runde Klammern gesetzt und an die Bezeichnung der Feldvariablen angehängt.

Einige Bezeichnungsbeispiele:

Un_{We} bzw. UnWe	Bezeichnet die dimensionslose Unrundheit der Welle.
$kant_{2Amp}$ bzw. kant2Amp	Bezeichnet die dimensionsbehaftete Amplitude der Komponente in Richtung der Achse 2 des Koordinatensystems 1-2-3 einer zeitvariablen Verkantung der Welle in der Lagerschale.
Kant_k	Bezeichnet den konstanten Eingabewert der in der Regel zeitabhängigen variablen Verkantung "Kant". Die Variable "Kant" ist im Programm ein Feld von N_T Werten, während Kant_k eine skalare Größe ist. An der Programmoberfläche wird der Wert der internen variablen Kant_k aber auch einfach mit Kant bezeichnet, um die Zugehörigkeit zu der physikalischen Erscheinung "Verkantung der Welle" hervorzuheben. Im Symbolverzeichnis werden die Variablen mit dem Zusatz "..._k" nicht extra aufgeführt.
Kant_0	Bezeichnet die Steuervariable zum Parameter "Kant", die angibt, ob der Parameter "Kant" ein primärer oder sekundäre Eingabe- oder Ergebnisparameter ist. Im Symbolverzeichnis werden die Variablen mit dem Zusatz "..._0" nicht extra aufgeführt.
$P(J_Z,J_X,J_T)$ bzw. P(JZ,JX,JT)	Bezeichnet den dimensionslosen Schmierfilmdruck an der Stelle J_Z, J_X, J_T des dreidimensionalen Feldes aller berechneten Schmierspaltdrücke. Es ist der Druck am Punkt $[J_Z,J_X]$ des über den Schmierspalt aufgespannten Gitternetzes zum Zeitpunkt J_T.
$P(N_Z,N_X,N_T)$ bzw. P(NZ,NX,NT)	Bezeichnet das gesamte Datenfeld der punktweise abgespeicherten Druckverteilungen über den Schmierspalt und über die Zeit für J_Z=1 bis N_Z, J_X=1 bis N_X und J_T=1 bis N_T.

Die Tabelle 5.2 im Anhang der Dokumentation enthält alle Parameterbezeichnungen der Dokumentation, der Programmoberfläche und der programminternen Variablen, die nach der erläuterten Nomenklatur im gesamten Programm einheitlich bezeichnet wurden. Die Tabelle 3.1 enthält die wesentlichen Basissymbole. Basissymbole, die grundsätzlich ohne Bezeichnungszusatz verwendet werden, sind in dieser Tabelle nicht aufgeführt. Ihre Bedeutung ist dem Symbolverzeichnis im Anhang zu entnehmen. Siehe Tabelle 5.2 im Anhang. Die Tabelle 3.2 enthält die verwendeten Bezeichnungszusätze. Die genaue Bedeutung der kompletten Symbolik ist dem Symbolverzeichnis (Tabelle 5.2 im Anhang) zu entnehmen.

Neben den Bezeichnungszusätzen, die üblicherweise an das Basissymbol angehängt werden, werden gelegentlich 4 Bezeichnungszusätze verwendet, die dem Basissymbol vorangestellt werden. Das sind die Bezeichnungszusätze Δ..., Delta... und D... . Sie stehen alle gleichbedeutend für die Differenz bzw. Schrittweite zweier Werte des nachfolgenden angegebenen Parameters. Diese verschiedenen Schreibweisen sind dem Umstand geschuldet, dass FORTRAN keine griechischen Buchstaben darstellen kann.

Im Quelltext der einzelnen Routinen sind weitere lokale Variable definiert. Ihre Bedeutung wird nur im Kopf der jeweiligen Routine erklärt. Ihre Schreibweise ist willkürlich und entspricht nicht der oben erläuterten Schreibweise. Die Bedeutung der globalen Variablen des einheitlichen Bezeichnungssystems wird in den einzelnen Routinen nicht noch einmal erklärt, da ihre Bedeutung zentral im Verzeichnis der Symbole (Anhang: 5.2 Symbolverzeichnis) erklärt ist.

Tabelle 3.1: *Verzeichnis der Basissymbole, die in verschiedenen Kombinationen mit Bezeichnungszusätzen verwendet werden*

Basissymbole		**Beschreibung**
a	A	Spaltfläche
-	A...	Koeffizienten des linearen bzw. linearisierten Reynoldsschen Differentialgleichung
-	a...	sonstige Koeffizienten (Hilfsvariablen für Zwischenergebnisse)
b...	B...	Lagerbreite, Breite eines Lagerabschnitts
ba...	Ba...	Betrag der Balligkeit der Welle oder der Lagerschale
-	Bez...	Liste von Bezeichnungen ... zu den bisher einprogrammierten Gerätetypen
bieg...	Bieg...	Betrag der Biegung der Welle
c	C	Mischungskonstante
-	C...	Hilfsvariablen
d...	-	Durchmesser

e...	E...	Exzentrizität
f...	(So...)	Lagerbelastung
-	F...	Füllungsgrad
freq	Freq...	Phasenfrequenz
h...	H...	Spalthöhe
-	J...	Laufindex
-	K...	Koeffizienten (Hilfsvariablen)
kant...	Kant...	Betrag der Wellenverkantung
ko...	Ko...	Konizität des Lagerschale
lei...	Lei...	Leistung
-	mm...	Platzhalter in COMMON-Blöcken für ganzzahlige Variablen, die in der entsprechenden Routine nicht benötigt werden
mo...	Mo...	Kippmoment an der Welle eines asymmetrischen Lagers
-	N...	Anzahl
-	nn...	Platzhalter in COMMON-Blöcken für ganzzahlige Variablen, die in der entsprechenden Routine nicht benötigt werden
p...	P...	Druck
-	Pa...	Feld der Parameter der Gerätevarianten
-	Pn...	vorläufiger Näherungswert für den Druck P
q...	Q...	Schmiermittelstrom
(f...)	So...	Sommerfeldzahl = dimensionslose Lagerbelastung (dimensionsbehaftete Lagerbelastung siehe f)
-	Symb...	Tabelle der Symbole der Parameter der bisher einprogrammierten Gerätevarianten
t...	T...	Zeit
-	Tol...	geforderte Genauigkeit bei der Lösung des linearen Gleichungssystems
un...	Un...	Betrag der Unrundheit der Welle bzw. Lagerschale
v...	-	Schmiermittelströmungsgeschwindigkeit
vol...	Vol...	Volumen
x...	X...	Umfangskoordinate des dimensionslosen X-Z-Koordinatensystems bzw. des dimensionsbehafteten x-y-z Koordinatensystems
-	xx...	Platzhalter in COMMON-Blöcken für reelle Variablen, die in der entsprechenden Routine nicht benötigt werden
y	-	radial zum Zentrum zeigende Koordinate des Lagerschalen-Koordinatensystem x-y-z
-	yy...	Platzhalter in COMMON-Blöcken für reelle Variablen, die in der entsprechenden Routine nicht benötigt werden
z...	Z...	axiale Koordinate des lagerschalenfesten Koordinatensystem x-y-z
-	zz...	Platzhalter in COMMON-Blöcken für reelle Variablen, die in der entsprechenden Routine nicht benötigt werden
η... eta...	Eta...	dynamische Viskosität der flüssigen Phase des Schmiermittels
φ... phi...	Φ... Phi...	Phasenwinkel
ω... omega...	Ω... Omega...	Winkelgeschwindigkeit der Welle bezogen auf das lagerschalenfeste Koordinatensystem x-y-z

***Tabelle 3.2**: Verzeichnis der Bezeichnungszusätze die in verschiedenen Kombinationen mit den Basissymbolen verwendet werden*

Bezeichnungszusatz	**Beschreibung**
..._0	Bezeichnungszusatz, der den sekundären Steuerparameter Xyz_0 zu einem Basisparameter Xyz bezeichnet. Dieser Steuerparameter Xyz_0 codiert die Information, ob der Basisparameter Xyz ein primärer, sekundärer oder irrelevanter Eingabe- oder Ergebnisparameter ist. Xyz_0 = -2 bedeutet: Xyz ist ein sekundärer Ergebnisparameter = -1 Xyz ist ein sekundärer Eingabeparameter = 0 Xyz ist ein irrelevanter Parameter = 1 Xyz ist ein primärer Eingabeparameter = 2 Xyz ist ein primärer Ergebnisparameter
..0, ..1, ..2, ..3, ...	Nummerierungen
...1, ...2, ...3	Achsenbezeichnungen der Komponenten bezogen auf das Koordinatensystem 1-2-3
..._Ak	aktueller skalarer Wert Xyz_Ak aus einem zeitlich variablen Parameterfeldes $Xyz(N_T)$. $Xyz\_Ak=Xyz(J_T)$, wenn J_T die Nummer des aktuellen Zeitpunktes ist.
...Amp	Amplitude
...Anf	Anfangswert
...Bieg	Wellenbiegung
D..., Delta...	Differenz bzw. Schrittweite (steht im Programm stellvertretend für das Zeichen Δ)
...Dim	dimensionsbehaftet
...E	Exzentrizität
...End	Endwert
...f	Kraft
...Fl	flüssige Phase des Schmiermittels
...Gas	gasförmige Phase des Schmiermittels
...Ge	Flüssigkeits-Gas-Gemisch
...geglättet	geglätteter Funktionsverlauf
...Ges	Gesamtwert, Summe
...Glei	Gleichung
...Chp	integrierte Elastizitätsmatrix

..._k	Bezeichnungszusatz eines skalaren Eingabeparameter Xyz_k, der einen aktuell zeitlich konstanten Wert eines eigentlich zeitlich variablen Parameterfeldes Xyz(N_T) enthält. An der Programmoberfläche wird dieser Bezeichnungszusatz nicht gezeigt.
..._k0	Kombination der Bezeichnungszusätze ..._k und ..._0
...Kant	Wellenverkantung
...La	Lagerschale
...Max	Maximalwert
...Min	Minimalwert
...Mit	Mittelwert
...Mo	Wellendrehmoment
...P	Druck
...Pa	Parameter der Gerätevarianten in den Verbindungsleitungen
...Pu	Pumpe
...Rand	Lagerrand bzw. Schmiertaschenrand
...Reib	Reibung
...S	Schnittpunkt
...So	Sommerfeldzahl, dimensionslose Lagerbelastung
...Spalt	Schmierspalt
...Sym	Symmetrie
...T, ...t	Zeit, im Quellcode und an der Programmoberfläche auch die Ableitung nach der Zeit, z.B. PT=∂P/∂T bzw. pt=∂p/∂t
...Ta	Schmiertasche
..._tics	Skalenteilung in der grafischen Darstellung mit GNUPLOT
...Typ	Gerätetyp
...Um	Umrechnungsfaktor eines dimensionslosen Parameters in den entsprechenden dimensionsbehafteten
...Var	Gerätevarianten
...Ve	Verbindungsleitung
...Ver	Verlust
...Vers	Versatz der Lagerabschnitte
...We	Welle
...Wechsel	Grenzwert zum Wechsel der Berechnungsmethode
...X, ...x	X-Richtung bzw. partielle Ableitung, z.B. $P_X=\partial P/\partial X$ bzw. $p_x=\partial p/\partial x$
...Y	Y-Richtung
...Z, ...z	Z-Richtung bzw. partielle Ableitung, z.B. $P_Z=\partial P/\partial Z$ bzw. $p_z=\partial p/\partial z$
Δ..., Delta..., D...,	Differenz bzw. Schrittweite z.B. $\Delta T(J_T)=T(J_T)-T(J_T-1)$

3.3.2 Das System der Steuerparameter

Mit dem Programm SIRIUS können qualitativ verschiedene Lagertypen, Betriebsbedingungen, theoretische Modelle und Berechnungsziele simuliert werden, zwischen denen der Anwender wählen kann und die zu einer großen Anzahl konkreter Varianten kombiniert werden können. Um das programmtechnisch zu organisieren, wurde neben den technischen Parametern, die die technischen Merkmale des Lagers quantifizieren, ein System von Steuerparametern implementiert, mit dem die jeweiligen qualitativen Merkmale der zu untersuchenden Variante codiert werden. Dementsprechend wurde für jedes qualitative Merkmal, für das es mindestens 2 Varianten gibt, ein Steuerparameter eingeführt. Die Steuerparameter sind generell als ganzzahlige Variablen vereinbart, auch wenn aktuell nur zwei Varianten vorgesehen sind. Damit können dem Programm relativ einfach weitere Varianten hinzugefügt werden. Anhand der aktuell geltenden Steuerparameter wird der Ablauf des gesamten Programms gesteuert. Das Programm ruft die erforderlichen Routinen auf und wählt die geeigneten Formeln aus. Eine für den Anwender besonders hilfreiche Funktion haben die Steuerparameter, indem mit ihrer Hilfe aus der großen Anzahl der möglichen technischen Eingabeparameter diejenigen ausgewählt werden, die für die aktuelle Variante relevant sind, so dass der Anwender nicht von einer großen Anzahl unsinniger Abfragen überhäuft wird.

Die Steuerparameter arbeiten im Wesentlichen unter der Programmoberfläche und der Anwender braucht sich um die Codierung keine Gedanken zu machen. Dort, wo qualitative Entscheidungen über die gewünschte Lagervariante zu treffen sind, kann in den entsprechenden Menüs zwischen den einzelnen verbal beschriebenen Varianten ausgewählt werden. Das Programm belegt dann den entsprechenden Steuerparameter mit dem zugehörigen Code. Die Codierung der Steuerparameter entspricht in der Regel den Aktionsnummern im Menü, mit denen die jeweilige Variante ausgewählt wird.

Die Steuerparameter zählen generell zu den Eingabedaten. Wenn sie zu den primären Eingabedaten gehören, wird der Anwender in einem entsprechenden Menü durch das Programm aufgefordert eine Variante auszuwählen. Es gibt aber auch eine große Anzahl sekundärer Steuerparameter, die aus der Eingabe der primären Steuerparameter abgeleitet werden. Diese sind in der Regel für den Anwender nicht sichtbar.

3.3.2.1 Die primären Steuerparameter des Hauptmenüs: "Festlegungen zur Theorie, zum Berechnungsverlauf und zum Lagertyp"

Die in der Eingabehierarchie ganz oben stehenden Steuerparameter sind die des 2.Hauptmenüs des PreProzessors "Festlegungen zur Theorie, zum Berechnungsverlauf und zum Lagertyp" (siehe dazu Bedienanleitung Abschnitt 4.4.2 und Unterabschnitte). Sie bestimmen im Wesentlichen die zu modellierende Lagervariante, die Betriebsbedingungen, das zu verwendende theoretische Modell und die daraus resultierenden weiteren erforderlichen Eingaben.

```
-----------------------------------------------------------------------
Festlegungen zur Theorie, zum Berechnungsverlauf und zum Lagertyp
-----------------------------------------------------------------------
-1- Erweiterte Reynoldssche Differentialgleichung          (Theo = 2)
-2- Belastung des Lagers vorgegeben                        (Last = 2)
-3- Vollstaendig umschlossenes Lager                      (Vollum = 1)
-4- Asymmetrisches Lager                                     (Sym = 2)
-5- Verkantete Welle innerhalb der Lagerschale            (Kante = 2)
-6- Gebogene Welle                                        (Biege = 2)
-7- Zwei versetzte Lagerabschnitte                       (Versatz = 2)
-8- Welle mit allen moeglichen Formabweichungen           (Welle = 4)
-9- Lagerschale mit allen möglichen Formabweichungen      (Schale = 6)
-10- Lagerschale bestehend aus gleichen Sektoren            (Sym3 = 2)
-11- Anzahl der gleichen Sektoren des Lagers               (NSym3 = 1)

-20- Evtl. einige Eingabeparameter zeitabhaengig          (Dynamic = 2)
-21- Zeitschritte DT variabel, Eingabe T(JT)            (SchrittVar = 2)
-22- Omega variabel, Eingabe Omega(JT)                    (OmegaVar = 2)
-23- Eingabe der Verlagerungsbahn E(JT), XE(JT)           (VerlagVar = 2)
-24- Belastung variabel, Eingabe f1(JT),f2(JT)            (LastVar = 4)
-25- Verkantung variabel, Eingabe Kant1(JT),Kant2(JT)     (KantVar = 4)
-26- Biegung variabel, Eingabe Bieg(JT),XBieg(JT)         (BiegVar = 3)
-30- Dimensionsbehaftete Ein- und Ausgabeparameter            (Dim = 2)
```

Die oben dargestellte Liste im Hauptmenü zeigt rechts die hier implementierten Steuerparameter und beispielhaft jeweils eine mögliche Belegung. Der Text in der jeweiligen Zeile beschreibt in Stichworten die Variante, die der aktuell dargestellten Code-Nummer entspricht. Die ausführliche Beschreibung der hiermit spezifizierbaren Varianten und ihre Codierung sind beschrieben in den Unterabschnitten 4.4.2.1 bis 4.4.2.19 der Bedienanleitung.

Programmintern sind diese Steuerparameter gemeinsam abgelegt im COMMON-Bock "common/Steuerparameter/", auf die alle Routinen bei Bedarf direkt zugreifen können.

3.3.2.2 Die sekundären Steuerparameter des COMMON-Blocks "/Datenregister/"

Um die Eingabe der zeitlich variablen Parameter und die Sicherung der Eingabe- und Ergebnisdaten programmtechnisch effektiv und übersichtlich zu gestalten, wurde jedem Parameter, der in irgendeiner Variante ein primärer Eingabe- oder Ergebnisparameter sein kann, ein sekundärer Steuerparameter zugeordnet, in dem dieses Merkmal codiert werden kann. Alle diese sekundären Steuerparameter sind in dem COMMON-Block "common/Datenregister/" abgelegt. Dabei gilt folgende Nomenklatur: Einem Parameter Xyz wird der Steuerparameter Xyz_0 zugeordnet. Das entsprechende Merkmal des Parameter Xyz wird im Steuerparameter Xyz_0 durch folgende Werte codiert:

Xyz_0 = 2 bedeutet: Xyz ist ein primärer Ausgabeparameter.

Xyz_0 = 1 bedeutet: Xyz ist ein primärer Eingabeparameter.

Xyz_0 = 0 bedeutet: Xyz ist ein irrelevanter Parameter.

Xyz_0 = -1 bedeutet: Xyz ist ein sekundärer Eingabeparameter.

Xyz_0 = -2 bedeutet: Xyz ist ein sekundärer Ausgabeparameter.

All den Variablen, die ausschließlich nur sekundäre Parameter sein können oder auch nur programmintern genutzt werden, wurde kein entsprechender Steuerparameter zugeordnet.

Die Belegung der Steuerparameter Xyz_0 mit Werten erfolgt durch die Routine "AktualRegister", wenn die Codierung in Abhängigkeit von der gewählten Lagervariante variieren kann. Wenn die Belegung des entsprechenden Steuerparameters unveränderlich ist, erfolgt die Belegung einmalig in der Routine "Anfangsdaten".

Diese sekundären Steuerparameter werden verwendet in den Routinen "VarPara" und "AusgabePara5" zur effektiven Selektion der jeweiligen primären Eingabe- und/oder Ausgabeparameter entsprechend ihres aktuellen Merkmals. Siehe dazu auch die Abschnitte 3.3.3 und 3.3.5.

3.3.2.3 Die Steuerfelder KX, KZ und NG zur Darstellung der Schmiertaschen und der Codierung der erforderlichen Differenzengleichung und Gleichungsnummern

Mit SIRIUS ist es möglich beliebige Anordnungen von Schmiertaschen und Schmiernuten zu simulieren. Um diese Flexibilität des Programms zu realisieren, sind die drei Datenfelder $KX(N_Z,N_X)$, $KZ(N_Z,N_X)$ und $NG(N_Z,N_X)$ erforderlich, die ausschließlich Steuerparameter enthalten. Von diesen 3 Feldern bekommt der Anwender nur das Feld KX zu Gesicht und zwar in dem Menü, wo er die Anordnung von Schmiertaschen eingibt. Programmintern füllt das Programm die Felder KX, KZ und NG mit weiteren sekundären Steuerparametern auf, die dem Anwender aber nicht angezeigt werden. Mit diesen Parametern steuert das Programm die Aufstellung der passenden Koeffizientenmatrix des linearen Gleichungssystems zur Berechnung des Schmierfilmdrucks. Die Aufgaben und die Arbeitsweise der Steuerfelder KX, KZ und NG sind ausführlich beschrieben in den Abschnitten 3.4.1.5 und 3.4.1.6 der Programmbeschreibung und die Arbeit mit dem Steuerfeld KX zur Eingabe von Schmiertaschen im Abschnitt 4.4.8 der Bedienanleitung.

3.3.2.4 Die Steuerfelder des Universal-Schmiermittel-Versorgungssystems

Auch die Gestaltung des peripheren Schmiermittel-Versorgungssystems, welches insbesondere für die Simulation hydrostatischer Lager benötigt wird, kann sehr komplex und variantenreich abgebildet werden. Dazu werden die 4 Felder mit Steuerparametern benötigt:

$Pu_{Var}(N_{Pu})$	-Betriebszustände der Schmiermittelpumpen	siehe Abschnitt 3.3.6.1
$N_{PaTyp}(N_{Typ})$	-Anzahlen der physikalisch-technischen Parameter der Gerätetypen	siehe Abschnitt 3.3.6.2
$Typ_{Var}(N_{Var})$	-Typnummern der Gerätevarianten	siehe Abschnitt 3.3.6.3
$Ve(3,N_{Ve})$	-Struktur des Wirkschaltplans des Schmiersystems	siehe Abschnitt 3.3.6.4

Eine Schlüsselfunktion hat hier das Steuerfeld $Ve(3,N_{Ve})$. Mit ihm werden die Verbindungen zwischen den Schmiermittelpumpen und den Schmiertaschen über die Verbindungsleitungen dargestellt. Außerdem wird hiermit angegeben, mit welcher Gerätevariante zur Regelung der Schmiermittelströme die einzelnen Verbindungsleitungen ausgerüstet sind. Die ausführlichen Beschreibungen erfolgen in den angeführten Abschnitten.

3.3.3 Das System der primären und sekundären Eingabe- und Ergebnisparameter

Eine Variable im Programm kann je nach aktuell simulierter Lagervariante Eingabeparameter, Ergebnisparameter oder für die aktuelle Berechnung irrelevant sein. Außerdem kann diese Variable ein primärer oder ein sekundärer Eingabe- bzw. Ergebnisparameter sein. Es gelten hier folgende Definitionen:

Ein **primärer Eingabeparameter** ist ein Parameter, der vom Anwender vorgegeben werden muss.

Ein **sekundärer Eingabeparameter** ist ein Parameter, der vom Programm für die Hauptrechnung als Eingabewert benötigt wird, aber nicht mehr vom Anwender eingegeben werden muss, weil er durch die Eingabe der primären Eingabeparameter bereits eindeutig bestimmt ist.

Ein **irrelevanter Parameter** ist eine im Programm implementierte Variable, deren Wert für die aktuell simulierte Lagervariante keinerlei Bedeutung hat, weder als Eingabe noch als Ergebnisparameter.

Primäre Ergebnisparameter sind alle die Daten, die durch die Hauptrechnung im Solver berechnet werden und aus denen sich dann alle anderen evtl. interessierenden Ergebnisse relativ einfach berechnen lassen.

Sekundäre Ergebnisparameter sind alle die Daten, die sich aus den Eingabeparametern und den primären Ergebnisparametern mit relativ geringem Aufwand ohne erneute Hauptrechnung berechnen lassen.

Die Unterscheidung der implementierten Variablen des Programms nach diesen Klassifizierungskriterien ist deswegen erforderlich, weil die Daten je nach Zugehörigkeit zu einer dieser Klassifizierungen an verschiedenen Stellen des Programms unterschiedlich behandelt werden. Dazu hier die wesentlichen Beispiele:

In den Menüs des PreProzessors werden ausschließlich primäre Eingabeparameter abgefragt.

Alle sekundären Eingabeparameter werden im PreProzessor nach der Eingabe der primären Eingabeparameter berechnet. Für die "technischen" Parameter wird das erledigt durch die Routine "Komplettieren" für die sekundären Steuerparameter erledigen das die Routinen "AktualRegister" und "Muster".

Bei der Eingabe der möglicherweise zeitlich variablen Eingabeparameter werden nur die abgefragt, die auch wirklich zeitlich variable sind und deshalb in Form einer Tabelle für jeden einzelnen Zeitpunkt manuell eingegeben oder aus einer Datei eingelesen werden müssen.

Bei der Sicherung der Eingabe- und Ergebnisdaten in einer Datei zur späteren Weiterbearbeitung, zur späteren Auswertung oder zur Archivierung werden nur die primären Eingabe- und Ergebnisparameter gespeichert, weil alle anderen Daten daraus leicht reproduziert werden können und so Speicherplatz gespart wird.

Alle sekundären Ergebnisparameter werden im PostProzessor erst dann erzeugt, wenn sie durch den Anwender im Rahmen der Auswertung der Ergebnisse über ein Untermenü abgefragt werden oder für eine grafische Darstellung eines Teilergebnisses angefordert werden. Diese sekundären Ergebnisse können dann evtl. zur Dokumentation oder zur sonstigen Weiterverarbeitung in externen Dateien gespeichert werden. Programmintern werden diese Daten jedoch nicht gespeichert, sondern bei jeder Anforderung neu berechnet.

Die Merkmale der Parameter, primärer Eingabe- oder Ausgabeparameter oder irrelevant zu sein, wird in einem Satz sekundärer Steuerparameter verwaltet, die in dem Common-Block "common/Datenregister/" abgelegt sind. Siehe dazu Abschnitt 3.3.2.2.

Einige Beispiele für Eingabe- bzw. Ergebnisparameter und die Codierung ihres Merkmals im zugehörigen Steuerparameter:

Der primäre Steuerparameter **Theo** gibt an, nach welcher Theorie gerechnet werden soll und wird immer vom Programm abgefragt. Deshalb ist der zugeordnete sekundäre Steuerparameter immer mit dem Wert **Theo_0 = 1** belegt. Die Belegung des Parameters Theo_0 erfolgt in der Routine "Anfangsdaten".

Der skalare Parameter $\mathbf{P_{Rand1}}$ enthält den Wert für den dimensionslosen Druck am Lagerrand. Dieser Parameter wird immer vom Programm abgefragt und ist deshalb immer ein primärer Eingabewert. Deshalb ist auch dieser zugeordnete Steuerparameter immer mit dem Wert $\mathbf{P_{Rand1\_0} = 1}$ belegt. Die Belegung des Parameters P_{Rand1_0} erfolgt in der Routine "Anfangsdaten".

Das Datenfeld $\mathbf{P(N_Z,N_X,N_T)}$ enthält die Werte für die Druckverteilung im Schmierspalt über die Zeit. Diese Daten sind immer das Ergebnis der Hauptrechnung im Solver und sind deshalb auch immer primäre Ausgabeparameter. Deshalb ist der zugeordnete Steuerparameter permanent mit dem Wert **P_0 = 2** belegt. Die Belegung des Parameters P_0 erfolgt in der Routine "Anfangsdaten".

Der skalare Parameter $\mathbf{Un_{La}}$ enthält den dimensionslosen Betrag der Unrundheit der Lagerschale über den Umfang. Wenn eine Unrundheit angenommen wird (Bedingung: Schale=2), dann ist der Parameter Un_{La} ein primärer Eingabeparameter und der zugeordnete Steuerparameter nimmt den Wert $\mathbf{Un_{La\_0} = 1}$ an. Wenn eine ideal zylindrische Lagerschale angenommen wird (Bedingung: Schale=1), dann wird keine Angabe für den Betrag der Unrundheit benötigt und der Parameter ist irrelevant. Dann nimmt der zugeordnete Steuerparameter den Wert $\mathbf{Un_{La\_0} = 0}$ an. Die Belegung des Parameters Un_{La_0} erfolgt in der Routine "AktualRegister".

ERLÄUTERUNG: *Wenn im Hauptmenü " Festlegungen zur Theorie, zum Berechnungsverlauf und zum Lagertyp" die Bedingung "Lagerschale ideal zylindrisch und starr" (Schale=1) festgelegt wurde, wird grundsätzlich angenommen, dass die Lagerschale einen ideal zylindrischen Querschnitt aufweist. Die Variable Un_{La} ist dann irrelevant und kann mit jedem beliebigen Wert belegt sein. Dieser Wert wird vom Programm nicht zur Kenntnis genommen. Diese Variable erscheint dann auch nicht an der Programmoberfläche und kann auch nicht bearbeitet werden.*

Wenn im selben Hauptmenü die Bedingung "Lagerschale nur mit Formabweichungsfunktionen" (Schale=2) festgelegt wurde, wird grundsätzlich angenommen, dass die Lagerschale von der ideal zylindrischen Form abweicht und der Betrag der Abweichung wird durch den Parameter Un_{La} festgelegt. Die Lagerschale kann in diesem Fall aber trotzdem einen ideal zylindrischen Querschnitt aufweisen, nämlich dann, wenn Un_{La}=0 gesetzt wird.

Beide Festlegungen führen zu dem gleichen Ergebnis, auch wenn sie aus verschiedenen Eingabedatensätzen hervor gehen.

Das Datenfeld $\mathbf{So(N_T)}$ enthält die Beträge der dimensionslosen Lagerbelastung über die Zeit. Wenn aus einer vorgegebenen Wellenverlagerung die Druckverteilung im Lager und daraus die resultierende Lagerbelastung berechnet wird (Bedingung: Last=1), ist der Parameter So ein primärer Ergebnisparameter und der zugeordnete Steuerparameter nimmt den Wert **So_0 = 2** an. Wenn aber aus einem vorgegebenen zeitlich variablen Kraftverlauf $So(N_T)$ die Verlagerungsbahn berechnet werden soll (Bedingung: Last=2 und Dynamic=2 und LastVar=2 oder =3), ist das Datenfeld $So(N_T)$ ein primärer Eingabeparameter und der zugeordnete Steuerparameter nimmt den Wert **So_0 = 1** an. Dann ist das Feld der Lagerbelastungen für jeden Zeitpunkt einzugeben. Wenn der Betrag der Lagerbelastung als zeitlich konstanter Wert vorgegeben werden soll (Bedingung: (Last=2 und Dynamic=1), fragt das Programm im Menü konstante Parameter den Wert für die konstante Lagerbelastung ab und speichert diesen zunächst in der Variablen So_k. Da der Solver intern eine vorgegebene Lagerbelastung immer aus dem Feld $So(N_T)$ abfragt, füllt der PreProzessor programmintern das Feld $So(N_T)$ mit dem Wert So_k. So wird das Feld $So(N_T)$ zu einem sekundären Eingabeparameter und der zugeordnete Steuerparameter nimmt den Wert $\mathbf{So_0 = -1}$ an. Die Belegung des Parameters So_0 erfolgt in der Routine "AktualRegister".

3.3.4 Arbeiten mit dimensionslosen und dimensionsbehafteten Parametern

In der hydrodynamischen Schmiertheorie hat es sich bewährt, mit dimensionslosen Größen zu arbeiten, basierend auf dem Ähnlichkeitsgesetz, das durch die Sommerfeldzahl **So** gegeben ist. Sie ist definiert durch

$$(2.401) \qquad So = \frac{S^2}{\eta \cdot \omega b} \cdot \frac{f}{B \cdot d^2}$$

Die Sommerfeldzahl stellt die dimensionslos gemachte Lagerbelastung eines hydrodynamisch geschmierten Gleitlagers dar. Analog der Lagerbelastung können auch alle anderen Lagerparameter, die nicht Bezugsparameter sind, in geeigneter Weise dimensionslos gemacht werden. Die Definitionen aller dimensionslosen Parameter und die sich daraus ergebenden dimensionslosen Gleichungen für die Parameter untereinander sind angegeben im Abschnitt 2.2 "Dimensionslose Darstellung des Lagers".

Ein Vorteil der Arbeit mit dimensionslosen Parametern und Gleichungen für die Programmierung besteht unter anderem darin, dass man mit reinen Zahlengleichungen im Programm operieren kann und sich dabei zunächst in keiner Weise um die später verwendeten Maßeinheiten der dimensionsbehafteten physikalischen Parameter kümmern muss.

Außer mit den 3 dimensionsbehafteten Bezugsparametern **Wellendurchmesser d**, **dynamische Viskosität** η und **Bezugswinkelgeschwindigkeit** ω**b** arbeitet das Programm intern ausschließlich mit dimensionslosen Werten. Der Anwender kann neben den dimensionslosen Daten aber auch mit den dimensionsbehafteten Daten an der Programmoberfläche arbeiten. Wird ein dimensionsbehafteter Wert an der Programmoberfläche eingegeben, wird er sofort in den entsprechenden dimensionslosen Wert umgerechnet und als solcher gespeichert. Analog werden alle angezeigten oder anderweitig ausgegebenen dimensionsbehafteten Daten erst zum Zeitpunkt ihrer Anzeige bzw. Ausgabe aus dem entsprechenden dimensionslosen Wert berechnet.

Mit welchen Daten an der Programmoberfläche gearbeitet wird, wird durch den primären Steuerparameter Dim festgelegt. Siehe dazu ausführlich in der Bedienanleitung Abschnitt 4.4.2.19.

3.3.5 Arbeiten mit zeitlich variablen Parametern

Zur Simulation eines Lagers berechnet das Programm SIRIUS die Verteilung des Schmierfilmdrucks und weiterer Lagerparameter immer über mehrere Zeitpunkte (mindestens 2 Zeitpunkte). Das tut es auch, wenn nur ein stationärer (zeitlich unveränderlicher) Zustand des Lagers simuliert werden soll.

Neben der größeren Anzahl zeitlich konstanter Parameter gibt es im Programm SIRIUS ca. 40 zeitlich variablen Parametern. Alle Parameter, die im Programm zeitlich variabel sein können, werden programmintern als Felder abgespeichert, in denen die Eingabe- bzw. Ergebnisdaten für jeden einzelnen Zeitpunkt der Berechnung abgespeichert werden.

Von den möglicherweise zeitlich variablen primären Eingabeparametern werden im konkreten Fall aber meist nur einige wenige tatsächlich auch als variabel angenommen und müssen für jeden einzelnen Zeitpunkt eingegeben werden. Um nicht auch noch die aktuell zeitlich konstanten Parameter für jeden Zeitpunkt eingeben zu müssen, wurde zu jedem zeitlich variablen Parameter Xyz, der evtl. primärer Eingabeparameter sein kann, ein entsprechender konstanter Parameter mit der Bezeichnung $Xyz_{\_k}$ definiert. In Abhängigkeit von der Festlegung der primären Steuerparameter Dynamic, SchrittVar, OmegaVar, VerlagVar, LastVar, KantVar und BiegVar fragt dann das Programm im Hauptmenü "Eingeben bzw. ändern der konstanten Parameter" den Parameter $Xyz_{\_k}$ bzw. im Hauptmenü "Eingeben bzw. ändern der zeitabhängigen (variablen) Parameter" das Feld des Parameters Xyz ab. Auf diese Weise kann der Aufwand der Dateneingabe oft erheblich reduziert werden. Falls ein potentiell variabler Parameter zunächst als konstanter Parameter $Xyz_{\_k}$ eingegeben wurde, füllt das Programm anschließend das Feld Xyz mit diesem Wert auf. Während der Hauptrechnung betrachtet das Programm alle potentiell zeitlich variablen Parameter als tatsächlich zeitlich variabel und greift ausschließlich auf die Felder der zeitlich variablen Parameter zu.

Die Felder der zeitlich variablen Parameter sind im COMMON-Block "common/variableParameter/" abgelegt. Die entsprechenden aktuell zeitlich konstanten Parameter sind im COMMON-Block "common/konstVarParameter/" als skalare Variablen abgelegt. Welcher dieser Parameter dann im entsprechenden Eingabemenü abgefragt und später im Ergebnisdatensatz ausgegeben wird, steuern die zugehörigen sekundären Steuerparameter des Common-Blocks /common/Datenregister/". Siehe dazu Abschnitt 3.3.2.2.

BEISPIEL: *Ist ein potentiell zeitlich variables Parameterfeld, z.B. Kant(N_T), in der aktuellen Lagervariante zeitlich konstant, wird der entsprechende konstante Eingabeparameter* $Kant_{\_k}$ *zum primären Eingabeparameter und im PreProzessor abgefragt. Dieses Merkmal ist codiert in dem zugehörigen sekundären Steuerparameter mit* $Kant_{\_k0}$*=1. Das Parameterfeld Kant(N_T) wird dann zum sekundären Eingabeparameter. Dieses Merkmal ist codiert in dem zugehörigen sekundären Steuerparameter mit* $Kant_{\_0}$*=-1.*

Ist das variable Parameterfeld, z.B. Kant(N_T) aktuell tatsächlich ein zeitlich variabler Parameter, wird dieser Parameter zum primären Eingabeparameter und im PreProzessor abgefragt. Dieses Merkmal ist codiert im zugehörigen sekundären Steuerparameter mit $Kant_{\_0}$*=1. Der entsprechende konstante Parameter* $Kant_{\_k}$ *wird dann irrelevant. Dieses Merkmal ist codiert im zugehörigen sekundären Steuerparameter* $Kant_{\_K0}$*=0.*

Es gibt noch einen weiteren COMMON-Block "common/aktuelleVarParameter/" zum Zwischenspeichern einzelner Werte zeitlich variabler Parameter. Hier werden die Werte der variablen Parameter zwischengespeichert, die für den aktuell bearbeiteten Zeitpunkt J_T gelten. Der aktuelle Wert des zeitlich variablen Parameters Xyz(J_T) wird mit Xyz_Ak bezeichnet.

Es gibt noch weitere Felder zeitlich variabler Parameter. Das sind die Felder $P(N_Z,N_X,N_T)$ für die Verteilung des Schmierfilmdrucks, $P_{Pu}(N_{Ta},N_T)$ für die Schmiertaschendrücke, $Q_{Pu}(N_{Pu},N_T)$ für die Pumpenölströme, $P_{Ta}(N_{Ta},N_T)$ für die Schmiertaschendrücke und $Q_{Ve}(N_{Ve},N_T)$ für Ölströme in den Verbindungsleitungen. Diese sind im COMMON-Block "common/DruckZeit/" abgelegt. Da diese Parameter immer Ergebnisparameter sind und damit immer zeitlich variabel, sind im Programm für diese Parameter keine zeitlich konstanten Variablen vereinbart.

3.3.6 Die Parameter zur Beschreibung des peripheren Universal-Schmiermittel-Versorgungssystems

Neben den Parametern, die das eigentliche Lager beschreiben, gibt es einen komplexen Datensatz aus Steuerparametern und technischen Parametern, die das periphere Universal-Schmiermittel-Versorgungssystem abbilden. Diese Parameter sind abgelegt in den Common-Blöcken "common/Schmierung/", "/common/Geraetevarianten/" und zum Teil in "common/DruckZeit/". Dem normalen Programmbenutzer bleibt das Studium dieser Datenstruktur im Wesentlichen erspart. Ihm präsentiert sich die Programmoberfläche in einer hoffentlich wesentlich leichter verständlichen Form:

```
------------------------------------------------------------------------
Universal-Schmiermittel-Versorgungssystem
------------------------------------------------------------------------
NPu =  3  Anzahl der Schmiermittelpumpen
NTa =  6  Anzahl der Schmiertaschen
NVe =  7  Anzahl der Verbindungsleitungen
NVar=  6  Anzahl der Geraetevarianten

P u m p e n :
     max.Pumpendruck   max. Oelstrom
JPu     pPuMax(JPu)     qPuMax(JPu)
 1      10.0000 MPa     2.0000 L/min
 2      10.0000 MPa     2.0000 L/min
 3      10.0000 MPa     2.0000 L/min

G e r a e t e v a r i a n t e n  in den Verbindungsleitungen:
1 Nur Kapillare bzw. Leitungswiderstand
1  1 ccp       =10000.0000 mm^-3         Widerstandsbeiwert
2 Kapillare und Rueckschlagventil
2  1 ccp       =10000.0000 mm^-3         Widerstandsbeiwert
3 Blende und Kapillare in Reihe
3  1 ccp       = 2000.0000 mm^-3         Widerstandsbeiwert
3  2 cbl       =    0.0500 (L/min)^2/MPa Blendenbeiwert
4 Blende, Kapillare und Rueckschlagventil in Reihe
4  1 ccp       = 2000.0000 mm^-3         Widerstandsbeiwert
4  2 cbl       =    0.0500 (L7min)^2/MPa Blendenbeiwert
5 Nur PM-Regler
5  1 q0        =    0.1500 L/min         Oelstrom durch Regler bei Taschendruck PTa=0
5  2 qP        =    0.3000 L/min         Theoretischer Oelstrom bei Taschendruck PTa=PP
5  3 pP        =   10.0000 MPa           Pumpendruck bei Aufnahme der Kennlinie
5  4 pS        =    1.0000 MPa           Differenz zwischen Pumpendruck PPu und Druck im Kennlinien-Scheitelpunkt S
5  5 eta0      =   50.0000 mPa*s         Dynamische Viskositaet am Eingang des PM-Reglers bei Aufnahme der Kennlinie
5  6 eta1      =   50.0000 mPa*s         Dynamische Viskositaet am Eingang des PM-Reglers im Betriebszustand
5    qP1       =    0.3000 L/min         Theoretischer Oelstrom bei Taschendruck PTa=PPu
5    cpm       =    0.0150 L/min/MPa     Anstieg der Kennlinie im aufsteigenden Ast
5    rpm       =    3.5088 MPa*min/L     Stroemungswiderstand des vollstaendig geoeffneten PM-Reglers
6 PM-Regler und Rueckschlagventil
6  1 q0        =    0.1500 L/min         Oelstrom durch Regler bei Taschendruck PTa=0
6  2 qP        =    0.3000 L/min         Theoretischer Oelstrom bei Taschendruck PTa=PP
6  3 pP        =   10.0000 MPa           Pumpendruck bei Aufnahme der Kennlinie
6  4 pS        =    1.0000 MPa           Differenz zwischen Pumpendruck PPu und Druck im Kennlinien-Scheitelpunkt S
6  5 eta0      =   50.0000 mPa*s         Dynamische Viskositaet am Eingang des PM-Reglers bei Aufnahme der Kennlinie
6  6 eta1      =   50.0000 mPa*s         Dynamische Viskositaet am Eingang des PM-Reglers im Betriebszustand
6    qP1       =    0.3000 L/min         Theoretischer Oelstrom bei Taschendruck PTa=PPu
6    cpm       =    0.0150 L/min/MPa     Anstieg der Kennlinie im aufsteigenden Ast
6    rpm       =    3.5088 MPa*min/L     Stroemungswiderstand des vollstaendig geoeffneten PM-Reglers

V e r b i n d u n g s l e i t u n g e n :
JVe    Nummer der Verbindungsleitung
JPu    Nummer der verbundenen Pumpe
JVar   Nummer der Geraetevariante in der Leitung
JTa    Nummer der verbundenen Schmiertasche
Typ    Nummer des Geraetetyps

JVe JPu  JVar  JTa Typ Bezeichnung des Geraetetyps
 1   1 >-- 2--< 2    2 Kapillare und Rueckschlagventil
 2   1 >-- 1--< 3    1 Nur Kapillare bzw. Leitungswiderstand
 3   2 >-- 5--< 3    3 Nur PM-Regler
 4   2 >-- 6--< 4    4 PM-Regler und Rueckschlagventil
 5   1 >-- 4--< 5    6 Blende, Kapillare und Rueckschlagventil in Reihe
 6   2 >-- 5--< 6    3 Nur PM-Regler
 7   3 >-- 3--< 6    5 Blende und Kapillare in Reihe
```

Das sind die primären Eingabedaten für das Schmiermittel-Versorgungssystem des Demonstrationsbeispiels "Demo21", dessen bereits recht komplexe Struktur grafisch im Bild 4.025 dargestellt ist.

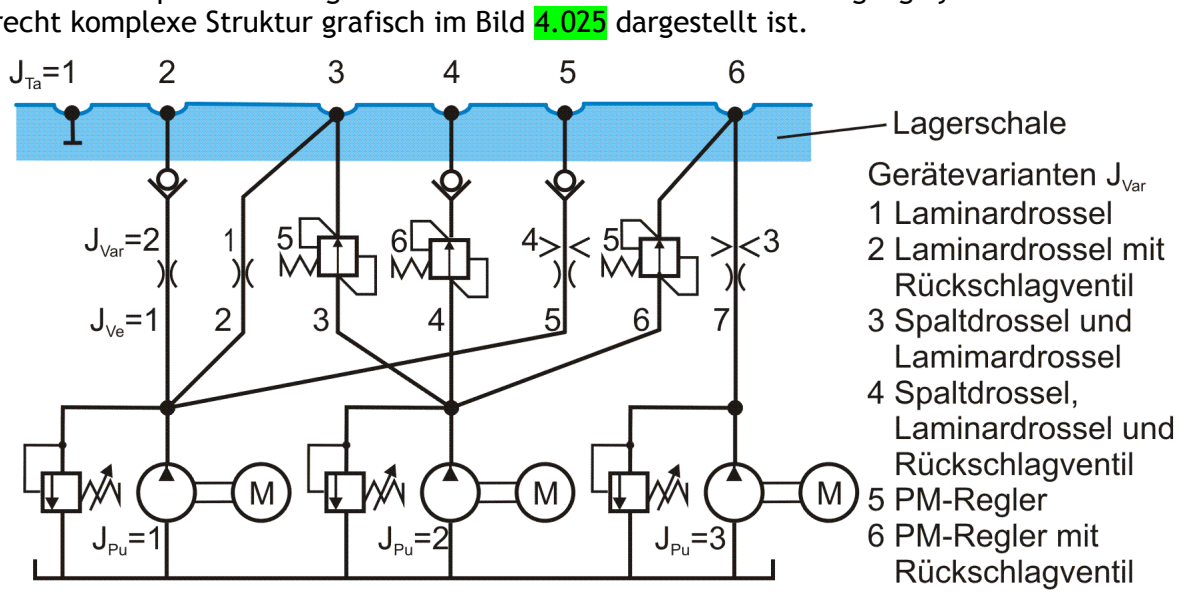


Bild 4.025: *Prinzipskizze einer möglichen Variante des externen Schmiermittel-Versorgungssystems*

Die Aktionen zur Eingabe werden ausführlich im Abschnitt 4.4.9 "Hauptmenu "Universal-Schmiermittel-Versorgungssystem" beschrieben. Wer aber evtl. das Programm erweitern will, z.B. durch einen weiteren Gerätetyp, der muss sich eingehend mit der Systematik dieser Datenstruktur vertraut machen, denn sie besitzt einige Besonderheiten. Deshalb wird das Datensystem in den nachfolgenden Unterabschnitten 3.3.6.1 bis 3.3.6.4 ausführlich erläutert.

3.3.6.1 Die Beschreibung der Schmiermittelpumpen

Die N_{Pu} Schmiermittelpumpen sind Konstantpumpen mit Druckbegrenzung (siehe Abschnitt 2.1.6.1). Um ihre idealisierten Betriebskennlinien darzustellen, werden lediglich die 2 Felder der maximalen Pumpendrücke $P_{PuMax}(N_{Pu})$ und der maximalen Pumpenölströme $Q_{PuMax}(N_{Pu})$ als primäre Eingabeparameter benötigt. Diese Parameter werden als zeitlich konstante Parameter

angenommen. Die tatsächlichen Pumpendrücke und -ölströme sind zeitlich variable primäre Ergebnisparameter und werden in den Feldern $P_{Pu}(N_{Pu},N_T)$ und $Q_{Pu}(N_{Pu},N_T)$ abgelegt.

Programmintern gibt es dann noch das **Feld von Steuerparametern $Pu_{Var}(N_{Pu})$**. In ihm ist codiert, auf welchem Ast der Kennlinie der aktuelle Betriebspunkt der jeweilige Pumpe gerade liegt (siehe Bild 3.14).

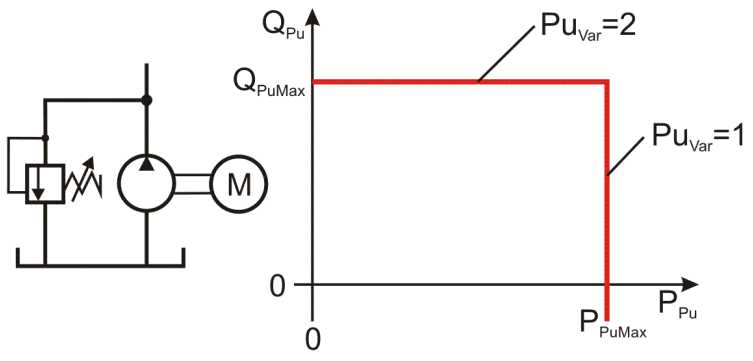


Bild 3.14: *Betriebskennlinie einer Schmiermittelpumpe*

Fördert die Pumpe J_{Pu} aktuell den maximalen Ölstrom in das Lager $Q_{Pu}(J_{Pu},J_T)=Q_{PuMax}(JPu)$, dann wird dieser Betriebszustand mit $Pu_{Var}(J_{Pu})=1$ codiert. Arbeitet die Pumpe aktuell als druckgeregelte Pumpe $P_{Pu}(J_{Pu},J_T)=P_{PuMax}(J_{Pu})$, dann wird dieser Betriebszustand mit $Pu_{Var}(J_{Pu})=2$ codiert. Mit Hilfe dieses Steuerparameters kann das Programm die geeignete Gleichung in die Koeffizientenmatrix für die Hauptrechnung schreiben.

3.3.6.2 Die verfügbaren Gerätetypen

Als Geräte werden hier die hydraulischen Bauteile bezeichnet, die in den Verbindungsleitungen zwischen den Pumpen und den Schmiertaschen angeordnet sind und den Schmiermittelstrom beeinflussen. Jeder Gerätetyp benötigt seine eigene Anzahl unterschiedlicher physikalisch-technischer Parameter zur Beschreibung seiner Wirkungsweise. Um nicht bei jeder Erweiterung einen neuen Satz von Variablen in das Programm einführen zu müssen, werden hier die Daten mit verschiedenster Bedeutung in einigen wenigen Datenfeldern gemeinsam abgelegt. Durch entsprechende Steuerfelder kann dann das Programm die Bedeutung der einzelnen Werte in den Feldern entschlüsseln und so die richtigen Gleichungen aufstellen.

Aktuell sind $N_{Typ}=6$ verschiedene Gerätetypen im Programm implementiert und es könnten noch weitere ergänzt werden. Die Namenbezeichnungen der Gerätetypen sind als Zeichenketten in dem **Feld $Bez_{Typ}(N_{Typ})$** fest abgelegt und die Anzahl der erforderlichen physikalisch-technischen Parameter zur Beschreibung des Geräteverhaltens in einem weiteren **Feld $N_{PaTyp}(N_{Typ})$**:

$$(3.002)\qquad Bez_{Typ} = \begin{pmatrix} \text{'Nur Kapillare bzw. Leitungswiderstand'} \\ \text{'Kapillare und Rueckschlagventil'} \\ \text{'Nur PM-Regler'} \\ \text{'PM-Regler und Rueckschlagventil'} \\ \text{'Blende und Kapillare in Reihe'} \\ \text{'Blende, Kapillare und Rueckschlagventil in Reihe'} \end{pmatrix} \qquad N_{PaTyp} = \begin{pmatrix} 1 \\ 2 \\ 9 \\ 9 \\ 2 \\ 2 \end{pmatrix}$$

Die Reihenfolge der abgelegten Bezeichnungen ist gleichzeitig die Nummerierung der Gerätetypen. Entsprechend dazu sind in 4 weiteren 2-dimensionalen Feldern die Zeichenketten zur Beschreibungen der einzelnen Parameter abgelegt:

Das **Feld $Bez_{Pa}(N_{Typ},N_{PaTyp})$** enthält alle Kurzbeschreibungen der einzelnen Parameter.

Das **Feld $Symb_{Pa}(N_{Typ},N_{PaTyp})$** enthält alle Symbole zur Bezeichnung der einzelnen Parameter in dimensionsloser Form.

Das **Feld $Symb_{PaDim}(N_{Typ},N_{PaTyp})$** enthält alle Symbole zur Bezeichnung der einzelnen Parameter in dimensionsbehafteter Form.

Das **Feld $Dim_{Pa}(N_{Typ},N_{PaTyp})$** enthält alle Maßeinheiten der einzelnen dimensionsbehafteten Parameter.

Z.B. hat das Feld $Symb_{Pa}$ folgenden fest vorgegebenen Inhalt:

$$(3.003)\qquad Symb_{Pa} = \begin{pmatrix} \text{'Ccp'} & & & & & & & & \\ \text{'Ccp'} & & & & & & & & \\ \text{'Q0'} & \text{'QP'} & \text{'PP'} & \text{'PS'} & \text{'Eta0'} & \text{'Eta1'} & \text{'QP1'} & \text{'Cpm'} & \text{'Rpm'} \\ \text{'Q0'} & \text{'QP'} & \text{'PP'} & \text{'PS'} & \text{'ETa0'} & \text{'Eta1'} & \text{'QP1'} & \text{'Cpm'} & \text{'Rpm'} \\ \text{'Ccp'} & \text{'Cbl'} & & & & & & & \\ \text{'Ccp'} & \text{'Cbl'} & & & & & & & \end{pmatrix}$$

Diese 6 Felder zur Beschreibung der aktuell implementierten Gerätetypen sind im Programm fest vorgegeben und können vom Anwender nur durch Zugriff auf den Quelltext verändert werden. Sie werden in der Routine "Anfangsdaten" definiert.

Warnung: *Es wird dringend davon abgeraten, die Reihenfolge der Gerätetypen zu verändern, weil sonst ältere Datensätze fehlinterpretiert werden. Auf diese Reihenfolge wird in verschiedenen Routinen aufgebaut, so dass eine Änderung dieser Reihenfolge leicht zu schwerwiegenden Programmfehlern führen kann.*

TIPP: *Sollte ein bereits implementierter Gerätetyp in einer verbesserten Version neu implementiert werden, ist es möglicherweise besser, diesen als einen neuen Typ hinten anzufügen. Die alte Version könnte dann wahlweise stillgelegt werden, indem sie aus den Eingabemenüs gestrichen wird, oder zur weiteren Nutzung mit alten Datensätzen weiterhin aktiv bleiben.*

Soll ein weiterer Gerätetyp in das Programm implementiert werden, brauchen nur diese Datenfelder und die Parameter N_{Typ} und evtl. N_{PaMax} erweitert werden und dem Programm stehen an der Nutzeroberfläche alle diese Daten zur Verfügung. Die eigentliche und wesentlich schwierigere Arbeit besteht dann allerdings darin, in die Routinen "KoMa4" bzw. "KoMa4_pack" die entsprechenden Befehle einzufügen, die anhand dieser Daten und der daraus abgeleiteten Gerätevarianten die richtigen Gleichungen in die Koeffizienten Matrix zur Hauptrechnung einfügen.

3.3.6.3 Die Beschreibung der Gerätevarianten

Als Gerätevarianten werden hier konkrete Gerät bezeichnet, die beschrieben werden durch einen Gerätetyp und einen dazu passenden Satz physikalisch-technischer Parameter. Es können maximal N_{TaMax} verschiedene Gerätevarianten vereinbart werden. In einem

Schmiersystem können Gerätevarianten verschiedenen Typs, aber auch mehrere Gerätevarianten gleichen Typs mit verschiedenen Werten der zugehörigen physikalischen Parameter vereinbart werden. Eine Gerätevariante kann in mehreren Verbindungsleitungen angeordnet werden. Die Nummerierung der Gerätevarianten bleibt dem Anwender überlassen. Sie ergibt sich aus der Reihenfolge der Eingabe, kann aber auch umsortiert werden. Siehe dazu Abschnitt 4.4.9.2.

Die Felder $\mathbf{Typ_{Var}(N_{TaMax})}$ und $\mathbf{Pa_{Var}(N_{TaMax},N_{PaMax})}$ zur Definition von $\mathbf{N_{Var}=6}$ Gerätevarianten haben für das hier gezeigte Demonstrationsbeispiel "Demo21" folgenden Inhalt:

$$(3.004)\qquad Typ_{Var} = \begin{pmatrix} 1 \\ 2 \\ 5 \\ 6 \\ 3 \\ 4 \\ \text{undef.} \\ \dots \end{pmatrix} \qquad Pa_{Var} = \begin{pmatrix} 0.31 & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{un} \\ 0{,}31 & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{un} \\ 0{,}06 & 13{,}58 & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{un} \\ 0{,}06 & 13{,}58 & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{un} \\ 1{,}52 & 3{,}06 & 3{,}82 & 0{,}38 & 1{,}00 & 1{,}00 & 3{,}06 & 0{,}40 & 0 \\ 1{,}52 & 3{,}06 & 3{,}82 & 0{,}38 & 1{,}00 & 1{,}00 & 3{,}06 & 0{,}40 & 0 \\ \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{undef.} & \text{un} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \end{pmatrix}$$

HINWEISE: *Die nicht benötigten Koeffizienten der Felder sind undefiniert, d.h. hier können beliebige Werte stehen, die vom Programm ignoriert werden.*

In dem Feld Pa_{Var} werden grundsätzlich nur die dimensionslosen Werte abgespeichert, deshalb unterscheiden sich diese Werte von den oben im Menü dargestellten Werten, die vor ihrer Anzeige in dimensionsbehaftete Werte umgerechnet wurden.

Die hier dargestellten Werte sind stark gerundet, um die Tabelle nicht zu groß werden zulassen.

3.3.6.4 Die Abbildung der Struktur des Wirkschaltplans des Schmiersystems

Die Gestaltung des peripheren Schmiermittel-Versorgungssystems kann sehr komplex und variantenreich abgebildet werden. Eine Schlüsselfunktion hat hier das Steuerfeld $Ve(3,N_{Ve})$. Mit ihm werden die Verbindungen zwischen den Schmiermittelpumpen und den Schmiertaschen über die Verbindungsleitungen dargestellt. Außerdem wird hier angegeben, mit welcher Gerätevariante zur Regelung der Schmiermittelströme die einzelnen Verbindungsleitungen ausgerüstet sind.

Der Anwender bekommt dieses Feld im Hauptmenü "Universal-Schmiermittel-Versorgungssystem" des PreProzessors in einer leichter verständlichen Form zu Gesicht. Der folgende Menüausschnitt zeigt die Daten dieses Feldes für das Demonstrationsbeispiel "Demo21":

```
V e r b i n d u n g s l e i t u n g e n :
JVe    Nummer der Verbindungsleitung
JPu    Nummer der verbundenen Pumpe
JVar   Nummer der Geraetevariante in der Leitung
JTa    Nummer der verbundenen Schmiertasche
Typ    Nummer des Geraetetyps

JVe JPu  JVar  JTa Typ Bezeichnung des Geraetetyps
 1   1 >-- 2--< 2    2 Kapillare und Rueckschlagventil
 2   1 >-- 1--< 3    1 Nur Kapillare bzw. Leitungswiderstand
 3   2 >-- 5--< 3    3 Nur PM-Regler
 4   2 >-- 6--< 4    4 PM-Regler und Rueckschlagventil
 5   1 >-- 4--< 5    6 Blende, Kapillare und Rueckschlagventil in Reihe
 6   2 >-- 5--< 6    3 Nur PM-Regler
 7   3 >-- 3--< 6    5 Blende und Kapillare in Reihe
```

Die Spalte J_{Pu} im Menü bildet die 1.Zeile des Feldes Ve, die Spalte J_{Ta} bildet die 2.Zeile und die Spalte J_{Var} die 3.Zeile. So hat das Steuerfeld $Ve(3,N_{Ve})$ für dieses Beispiel folgendes Aussehen:

$$(3.005)\qquad Ve = \begin{pmatrix} 1 & 1 & 2 & 2 & 1 & 2 & 3 \\ 2 & 3 & 3 & 4 & 5 & 6 & 6 \\ 2 & 1 & 5 & 6 & 4 & 5 & 3 \end{pmatrix}$$

Mit Hilfe dieser Matrix kann im Programm gezielt auf bestimmte Werte in Feldern zugegriffen werden, deren Index nicht explizit bekannt ist. So haben im Programmcode z.B. folgende Ausdrücke folgende Bedeutungen:

$x = P_{Pu}(Ve(1,7))$ bedeutet: x ist der Druck der Pumpe, die mit der 7.Verbindungsleitung verbunden ist. Das Programm ruft in diesem Beispiel den Druck der Pumpe Nr. 3 auf. D.h. $x=P_{Pu}(J_{Pu}=3)$

$x = P_{Ta}(Ve(2,7))$ bedeutet: x ist der Druck in der Schmiertasche, die ebenfalls mit der 7.Verbindungsleitung verbunden ist, welche in diesem Beispiel die Schmiertasche 6 ist. Damit sind also die Pumpe Nr.3 mit der Schmiertasche Nr.6 über die Leitung Nr. 7 miteinander verbunden.

$j = Typ_{Var}(Ve(3,7))$ bedeutet: In der Verbindungsleitung Nr. 7 ist ein Stromregelgerät des Typs Nr. 3 angeordnet. Das sind nach der programminternen Nummerierung der Gerätetypen eine Blende und eine Kapillare in Reihenschaltung. Das Programm ermittelt also die Nummer des Gerätetyps der Gerätevariante, die in die Leitung Nr. 7 eingebaut ist und speichert sie in j. Die Nummer der Gerätevariante $J_{Ve}=Ve(3,7)$ wird dabei explizit nicht ermittelt.

So beschreibt das oben gezeigte Beispiel des Steuerfeldes Ve in maschinenlesbarer Form ein peripheres Schmiermittel-Versorgungssystem, das eine Struktur gemäß Bild 4.025 aufweist.

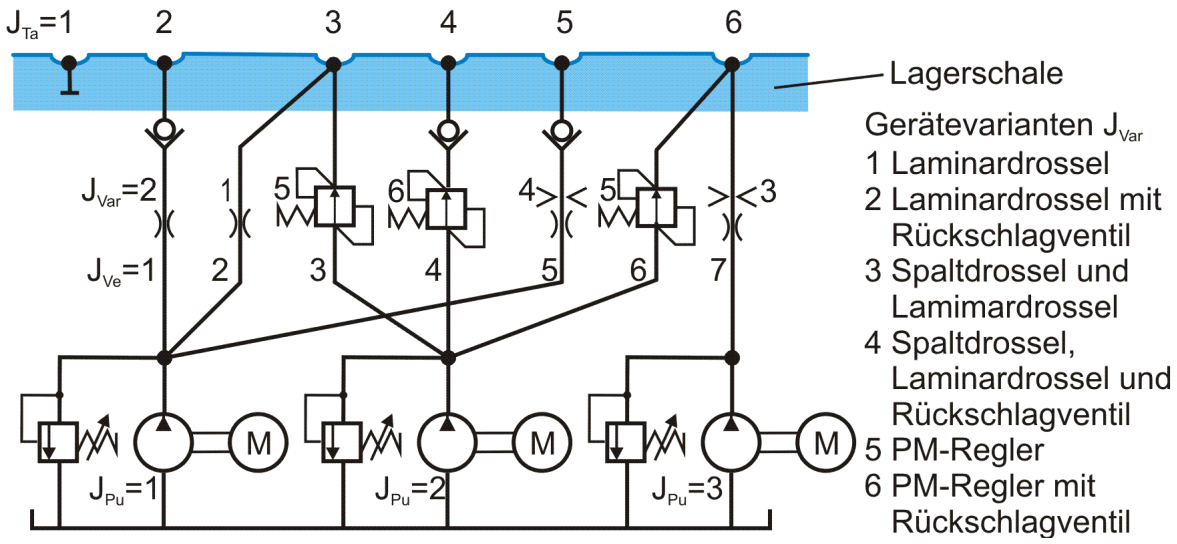


Bild 4.025: *Prinzipskizze einer möglichen Variante des peripheren Universal-Schmiersystems*

Der normale Anwender braucht sich auch um dieses Steuerfeld keine Gedanken zu machen. Die Kenntnis dieses Codierungsinstruments wird erst interessant, wenn man einen weiteren Gerätetyp in das Programm implementieren möchte.

3.3.7 Zusammenfassung der globalen Programmparameter in Common-Blöcken

Um den Überblick über alle im Programm SIRIUS verwendeten Variablen zu behalten, wurden grundsätzlich alle verwendeten Variablen am Anfang jeder Routine durch eine Format-Anweisung spezifiziert, obwohl die Programmiersprache FORTRAN das nicht zwingend vorschreibt. Außerdem wurden alle globalen Variablen in COMMON-Blöcke zusammengefasst abgelegt. Globale Variablen sind hier Daten, die in mehreren Routinen benötigt werden. In SIRIUS werden sie in den verschiedenen Routinen auch gleichlautend bezeichnet. Auch das wird nicht von der Programmiersprache gefordert. Es erleichtert aber erheblich die Lesbarkeit des Quellcodes.

COMMON-Blöcke sind eine spezielle Art der Vereinbarung von Speicherbereichen in FORTRAN, auf die verschiedene Routinen direkt zugreifen können. Ursprünglich stammt diese Idee aus der Zeit, als Speicherplatz noch sehr knapp war und so der gleiche Speicherplatz in verschiedenen Routinen mehrfach genutzt werden konnte. Hier werden die COMMON-Blöcke dazu genutzt, auf eine gemeinsame Datenbasis durch die verschiedenen Routinen direkt zuzugreifen. Dadurch brauchen globale Variablen, die in einer mehrfach untergeordneten Routine benötigt werden, nicht durch alle übergeordneten Routinen geschleust werden. Dabei wurden Datenblöcke nach funktionalen Kriterien zusammengefasst, so dass die Daten der einzelnen Blöcke in etwa in den gleichen Routinen benötigt werden, was natürlich nicht generell der Fall ist.

In den einzelnen Routinen werden nur die COMMON-Blöcke spezifiziert, die auch benötigt werden und hier wiederum nur die Variablen mit ihrer einheitlichen Bezeichnung, auf die die Routine zugreift. Da alle Variablen eines Common-Blocks immer in der gleichen Reihenfolge lückenlos aufgeführt werden müssen, werden aktuell nicht benötigte Variablen mit Dummy-Namen aufgeführt in der Art xx1,xx2,...yy1,yy2,...,nn1,nn2,...,mm1,mm2,..., so dass in jeder Routine sofort zu erkennen ist, welche Daten des jeweiligen COMMON-Blocks in der aktuellen Routine benötigt werden.

Beispiel:

Vollständige Belegung des Common-Bocks "Steuerparameter"

```
      common/Steuerparameter/
     * Status,Theo,Last,Vollum,Sym,Welle,Schale,Kante,Biege
     *,Dynamic,SchrittVar,OmegaVar,VerlagVar,LastVar,KantVar,BiegVar
     *,Dim,Versatz,Sym3,NSym3
```

Aktuelle Belegung in der Routine "FilmDruck1"

```
      common/Steuerparameter/
     * Status,Theo,mm3,Vollum,Sym,Welle,Schale,Kante,Biege
     *,mm11,mm12,mm13,mm14,mm15,mm16,mm17
     *,mm18,Versatz
```

Zur Arbeit mit COMMON-Blöcken siehe [8, Kapitel 6.1]. Um einen Überblick über alle in SIRIUS vereinbarten COMMON-Blöcke und die Reihenfolge der darin enthaltenen Parameter zu erhalten, sind alle Blöcke mit ihrem vollständigen Inhalt im Hauptprogramm "SIRIUS" aufgelistet.

Folgende COMMON-Blöcke wurden vereinbart:

common/Text/

Enthält die 3 Textvariablen "Datum", "Titel" und "Version".

common/Steuerparameter/

Enthält alle primären Steuerparameter, die im PreProzessor im Hauptmenü "Festlegungen zur Theorie, zum Berechnungsverlauf und zum Lagertyp" festgelegt werden. Außerdem ist hier der sekundäre Steuerparameter "Status" abgelegt.

common/Steuerfelder/

Enthält die Steuerfelder "KX", "KZ" und "NG". Im Feld KX ist die Anordnung der Schmiertaschen codiert. In den Feldern KX, KZ sind außerdem die Nummern der zu verwendenden Differenzengleichungen codiert, die zur Aufstellung des linearen Gleichungssystems zur Berechnung der Druckverteilung im Lager benötigt werden. Im Feld NG sind die Nummerierung der Gleichungen des Gleichungssystems abgelegt. Mit N_{Glei} ist hier außerdem die Anzahl der Gleichungen des Gleichungssystems angegeben.

common/Datenregister/

Enthält alle sekundären Steuerparameter vom Typ Xyz_0, die die zugehörigen Parameter Xyz als primäre oder sekundäre Eingabe- oder Ergebnisparameter oder als irrelevante Parameter klassifizieren. Siehe dazu Abschnitt 3.3.3.

common/Anzeigeauswahl/

Enthält nur das Steuerfeld "Anzeigen", mit dem auch die Auswahl der anzuzeigenden variablen Parameter gesteuert wird. Die manuelle Auswahl erfolgt mit der Aktion -92- im Hauptmenü des PostProzessors und wird durch die Routine "AuswahlAnzeige" ausgeführt.

common/interneParameter/

Enthält die internen Hilfswerte, die die nummerische Rechnung beeinflussen und vom Anwender möglichst nicht verändert werden sollen, aber trotzdem im Untermenü "Bearbeiten der programminternen Parameter" ohne Änderung des Quelltextes verändert werden können.

common/konstanteParameter/

Enthält zeitlich konstante Eingabe- und Ergebnisparameter.

common/VersatzLager/

Enthält die technischen Parameter, die für die Sondervariante "Versetzte Lagerabschnitte" (Steuerparameter: Versatz=2 oder =3) benötigt werden. Siehe dazu die Abschnitte 2.1.2.16 bzw. 2.2.2.16 und 4.4.2.7 und 4.4.4.11.

common/Belastungsfunktionen/

Enthält die Parameter, die benötigt werden, um einige zeitabhängige Eingabeparameter nicht punktweise eingeben zu müssen, sondern sie mit wenigen Daten als analytisch gegebene Funktionen zu definieren, um so die Eingabe zu erleichtern. Das betrifft die Funktionen der Verlagerungsbahn, des zeitlichen Verlaufs der Lagerbelastung, der Wellenverkantung, der Wellenbiegung und einer pendelnden Drehbewegung der Welle.

common/SpaGeo/

Enthält die Ergebnisse der Berechnung der Spaltgeometrie für den aktuellen Zeitpunkt J_T.

common/ElastVerform/

Enthält die erforderlichen Eingabe- und Ergebnisparameter für die Berücksichtigung elastischer Verformung der Lagerschale infolge des Schmierfilmdrucks.

HINWEIS: *Für diese Variante ist zwar inzwischen die erforderliche Datenstruktur im Programm SIRIUS implementiert. Die Berechnung ist aber noch nicht freigegeben, da bisher noch kein zuverlässiger und stabiler Berechnungsalgorithmus gefunden wurde.*

common/Schmierung/

Enthält Parameter zur Definition des peripheren Schmiermittel-Versorgungssystems.

common/Gerätevarianten/

Enthält ebenfalls Parameter zur Definition des peripheren Schmiermittel-Versorgungssystems, speziell die Parameter zur Beschreibung der stromregelnden Geräte in den Verbindungsleitungen.

common/variableParameter/

Enthält zeitlich variable Lagerparameter.

common/konstVarParameter/

Enthält analog zu den Feldern Xyz(N_T) des COMMON-Blocks "common/variableParameter/" die skalaren Variablen Xyz$_k$, in denen bei Bedarf die entsprechenden konstanten Werte eingegeben werden können. Ausführlich dazu siehe Abschnitt 3.3.5.

common/aktuelleVarParameter/

Dient als Zwischenspeicher für die variablen Parameter Xyz$_{Ak}$= Xyz(J_T) des aktuellen in der Berechnung befindlichen Zeitpunktes J_T.

common/Druck/

Dient als Zwischenspeicher für die aktuellen variablen Ergebnisparameter $P_{Ak}(N_Z,N_X)=P(N_Z,N_X,J_T)$, $H_{Ak}(N_Z,N_X)$, $FH_{Ak}(N_Z,N_X)$ und $P_{T\_Ak}(N_Z,N_X)$ des aktuellen in der Berechnung befindlichen Zeitpunktes J_T.

common/DruckZeit/

Enthält die kompletten Felder der primären, zeitlich variablen Ergebnisparameter $P(N_Z,N_X,N_T)$, $P_{Pu}(N_{Pu},N_T)$, $Q_{Pu}(N_{Pu},N_T)$, $P_{Ta}(N_{Ta},N_T)$ und $Q_{Ve}(N_{Ve},N_T)$.

common/Bezugsparameter/

Enthält die 4 Bezugsparameter Durchmesser d, relatives Lagerspiel S, dynamische Viskosität η und Bezugswinkelgeschwindigkeit ωb, mit denen die dimensionslosen Parameter in dimensionsbehaftete umgerechnet werden können und umgekehrt. Außerdem sind die daraus berechneten, aktuellen Umrechnungsfaktoren für die jeweiligen Parameter enthalten.

HINWEIS: *Der 5.Bezugsparameter "relative Lagerbreite" B ist im COMMON-Block "common/konstanteParameter/" abgelegt.*

3.4 Beschreibung der Lösungsmethoden der wesentlichen Teilaufgaben der numerischen Berechnungen

In den folgenden Unterabschnitten werden die wichtigsten, verwendeten nummerischen Lösungsmethoden unabhängig von ihrer Aufteilung auf die verschiedenen Routinen des Quelltextes des Programms erläutert.

3.4.1 Die Berechnung der Druckverteilung im Schmierspalt mit Hilfe des Differenzenverfahrens

Die Berechnung der Druckverteilung P(Z,X) im Schmierspalt bildet das Kernproblem der Berechnungen eines Gleitlagers. Diese Berechnung ist zu jedem Zeitpunkt J_T mindestens einmal auszuführen. Dazu ist die Reynoldssche Differentialgleichung zu lösen. Weder für die klassische Reynoldssche Differentialgleichung, noch für ihre Erweiterung gibt es eine analytische Lösungsmethode. Deshalb muss ein numerisches Verfahren angewendet werden, welches eine Näherungslösung erzeugt.

3.4.1.1 Das Prinzip des Differenzenverfahrens

Das Differenzenverfahren ist ein numerisches Verfahren, mit dem man lineare partielle Differentialgleichungen lösen kann. In unserem Fall ist eine lineare Differentialgleichung der Form

$$(3.006) \qquad \frac{\partial^2 P}{\partial X^2} + A_1 \cdot \frac{\partial^2 P}{\partial Z^2} + A_2 \cdot \frac{\partial P}{\partial X} + A_3 \cdot \frac{\partial P}{\partial Z} + A_4 \cdot P = R$$

gegeben. Dabei ist P(Z,X) die gesuchte Funktion, über eine begrenzte Fläche der X-Z-Ebene, dem Definitionsbereich der Funktion. Die **Koeffizienten A_1, A_2, A_3, A_4** und die **rechte Seite R** können berechenbare Funktionen von X und Z sein.

Am Rand des Definitionsbereiches sind Randbedingungen gegeben, die dafür sorgen, dass es genau eine Lösung für das Problem innerhalb der begrenzten Fläche gibt. In unserem Fall sind das der Umgebungsdruck an den Lagerrändern und die Drücke in den Schmiertaschen. Deshalb handelt es sich hier um ein Randwertproblem.

Das Differenzenverfahren funktioniert in der Weise, dass der Definitionsbereich mit einem diskreten Gitternetz überzogen wird und die Funktion P(Z,X) nur an den Gitterpunkten dieses Netzes berechnet wird. Die Tabelle dieser Werte stellt dann eine diskretisierte Näherungslösung dieses Problems dar.

Um dieses Problem numerisch lösbar zu machen, werden die partiellen Differentialquotienten $\partial^2P/\partial X^2$, $\partial P/\partial X$, $\partial^2P/\partial Z^2$ und $\partial P/\partial Z$ näherungsweise durch Differenzenquotienten ersetzt. Dazu wird die Lösungsfunktion P(Z,X) jeweils im Bereich der zu berechnenden Stelle durch ein Polynom approximiert, dass sich auf dem gesuchten Gitterpunkt Z,X und mindesten zwei weiteren Nachbarpunkten abstützt. Durch drei auf einer Linie liegende Gitterpunkte ist eine quadratische Funktion der Form

$$(3.007) \qquad P(X) \approx a_1 \cdot X^2 + a_2 \cdot X + a_3$$

eindeutig bestimmt. Außerdem sind damit auch ihre beiden Ableitungen bestimmt durch

$$(3.008) \qquad \frac{\partial P}{\partial X} = 2 \cdot a_1 \cdot X + a_2$$

und

$$(3.009) \qquad \frac{\partial^2 P}{\partial X^2} = 2 \cdot a_1$$

Daraus lassen sich dann die Formeln der benötigten Differenzengleichungen herleiten. Je nachdem, ob die drei Punkte mit konstanter oder variabler Schrittweite aufeinander folgen, ergeben sich verschiedene Differenzengleichungen. Im Programm SIRIUS werden die konstanten Schrittweiten ΔX und ΔZ verwendet. An den Lagerrändern und an den Rändern der Schmiertaschen ist der Abstand der Stützstellen aber nur ΔX/2 bzw. ΔZ/2. Deshalb werden im Programm die 3 Basisvarianten der entsprechenden Differenzengleichungen benötigt:

Basisvariante 1: Approximation über 3 Stützstellen mit der äquidistanten Schrittweite ΔX gemäß Bild 3.16.

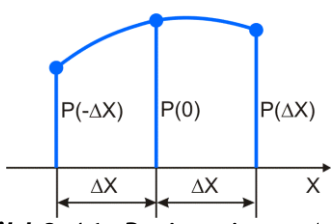


Bild 3.16: Basisvariante 1

$$(3.010) \qquad \frac{\partial P(0)}{\partial X} \approx \frac{-P(-\Delta X) + P(\Delta X)}{2 \cdot \Delta X}$$

$$(3.011) \qquad \frac{\partial^2 P(0)}{\partial X^2} \approx \frac{P(-\Delta X) - 2 \cdot P(0) + P(\Delta X)}{\Delta X^2}$$

Basisvariante 2: Approximation über 3 Stützstellen mit den 2 Schrittweiten ΔX/2 und ΔX gemäß Bild 3.17.

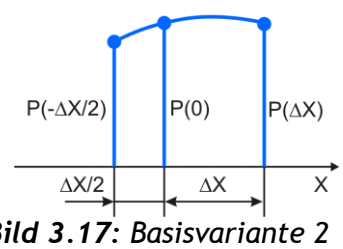


Bild 3.17: Basisvariante 2

$$(3.012) \qquad \frac{\partial P(0)}{\partial X} \approx \frac{-4 \cdot P(-\Delta X/2) + 3 \cdot P(0) + P(\Delta X)}{3 \cdot \Delta X}$$

$$(3.013) \qquad \frac{\partial^2 P(0)}{\partial X^2} \approx \frac{8 \cdot P(-\Delta X/2) - 12 \cdot P(0) + 4 \cdot P(\Delta X)}{3 \cdot \Delta X^2}$$

Basisvariante 3: Approximation über 3 Stützstellen mit den 2 Schrittweiten ΔX und ΔX/2 gemäß Bild 3.18.

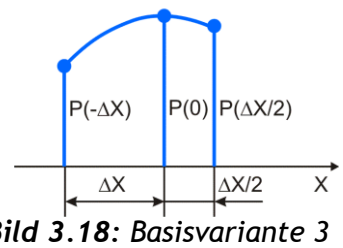


Bild 3.18: Basisvariante 3

$$(3.014) \qquad \frac{\partial P(0)}{\partial X} \approx \frac{-P(-\Delta X) - 3 \cdot P(0) + 4 \cdot P(\Delta X/2)}{3 \cdot \Delta X}$$

$$(3.015) \qquad \frac{\partial^2 P(0)}{\partial X^2} \approx \frac{4 \cdot P(-\Delta X) - 12 \cdot P(0) + 8 \cdot P(\Delta X/2)}{3 \cdot \Delta X^2}$$

Die Differenzengleichungen in Z-Richtung werden analog gebildet.

Z.B. für den Fall, dass für den aktuell untersuchten Punkt P(0,0) und seine benachbarten Punkte die äquidistanten Schrittweiten ΔX und ΔZ gelten, kann die lineare Differentialgleichung (3.006) durch die lineare Differenzengleichung approximiert werden.

$$(3.016) \qquad \left(\frac{A_1}{\Delta Z^2} - \frac{A_4}{2 \cdot \Delta Z}\right) \cdot P(-\Delta Z,0) +$$
$$+\left(\frac{1}{\Delta X^2} - \frac{A_3}{2 \cdot \Delta X}\right) \cdot P(0,-\Delta X) -$$
$$-\left(\frac{2}{\Delta X^2} + \frac{A_1 \cdot 2}{\Delta Z^2}\right) \cdot P(0,0) +$$
$$+\left(\frac{1}{\Delta X^2} + \frac{A_3}{2 \cdot \Delta X}\right) \cdot P(0,\Delta X) +$$
$$+\left(\frac{A_1}{\Delta Z^2} + \frac{A_4}{2 \cdot \Delta Z}\right) \cdot P(\Delta Z,0) = R$$

So lässt sich für jeden Gitterpunkt innerhalb des Definitionsbereichs eine lineare Gleichung aufstellen und es entsteht ein lineares Gleichungssystem, das sich mit einem geeigneten Lösungsverfahren für lineare Gleichungssysteme lösen lässt.

3.4.1.2 Lösung der klassischen Reynoldsschen Differentialgleichung

Die dimensionslose klassische Reynoldssche Differentialgleichung (siehe Abschnitt 2.2.3.1) ist eine lineare partielle Differentialgleichung 2.Ordnung.

$$(2.600) \qquad 0 = \frac{\partial^2 P}{\partial X^2} + \frac{1}{B^2} \cdot \frac{\partial^2 P}{\partial Z^2} + \frac{3}{H} \cdot \left(\frac{\partial H}{\partial X} \cdot \frac{\partial P}{\partial X} + \frac{1}{B^2} \cdot \frac{\partial H}{\partial Z} \cdot \frac{\partial P}{\partial Z}\right) - \frac{6}{\pi \cdot H^3} \cdot \left(\frac{\Omega}{2} \cdot \frac{\partial H}{\partial X} + \frac{\partial H}{\partial T}\right)$$

Sie hat die lineare Struktur

$$(3.018) \qquad \frac{\partial^2 P}{\partial X^2} + A_1 \cdot \frac{\partial^2 P}{\partial Z^2} + A_2 \cdot \frac{\partial P}{\partial X} + A_3 \cdot \frac{\partial P}{\partial Z} + A_4 \cdot P = R$$

Die Randbedingungen sind die Drücke an den Schmierspalträndern und den Schmiertaschenrändern.

HINWEIS: *Die "Gümbelsche Randbedingung", die besagt, dass negative Werte des berechneten Drucks anschließend auf Null gesetzt werden, ist in diesem Sinne keine echte Randbedingung. Zunächst wird der Druck auch hier über die gesamte Spaltfläche berechnet und im Anschluss daran werden erst die negativen Werte auf Null gesetzt. Das erfolgt durch die Routine "PKorr1".*

Das Differenzenverfahren ist hierfür geeignet und das Problem kann damit direkt gelöst werden. Die Felder der Koeffizienten $A_1(N_{Glei})$ bis $A_4(N_{Glei})$ und die rechten Seiten $R(N_{Glei})$ für jedes Flächenelement (J_Z,J_X) des eigentlichen Schmierspalts werden durch die Routine "KeoDGL3" berechnet.

HINWEIS: *Die Flächen der Schmiertaschen zählen nicht zum eigentlichen Schmierspalt.*

Die Berechnung des Druckverlaufs im Schmierspalt mit der klassischen Reynoldsschen Differentialgleichung wird durch die Routine "FilmDruck1" ausgeführt.

3.4.1.3 Linearisierung der erweiterten Reynoldsschen Differentialgleichung

Die dimensionslose erweiterte Reynoldssche Differentialgleichung (Abschnitt 2.2.3.2) ist nicht mehr linear.

$$(2.607)\qquad 0 = \frac{\partial^2 P}{\partial X^2} + \frac{1}{B^2}\cdot\frac{\partial^2 P}{\partial Z^2} + \frac{3}{H}\cdot\left(\frac{\partial H}{\partial X}\cdot\frac{\partial P}{\partial X} + \frac{1}{B^2}\cdot\frac{\partial H}{\partial Z}\cdot\frac{\partial P}{\partial Z}\right) - \frac{6}{\pi\cdot H^3}\cdot\frac{P}{P+C}\cdot\left(\frac{\Omega}{2}\cdot\frac{\partial H}{\partial X} + \frac{\partial H}{\partial T}\right) - \frac{6}{\pi\cdot H^2}\cdot\frac{C}{(P+C)^2}\cdot\left(\frac{\Omega}{2}\cdot\frac{\partial P}{\partial X} + \frac{\partial P}{\partial T}\right)$$

Sie ist eine nichtlineare partielle Differentialgleichung 2.Ordnung. Bezogen auf die Ortskoordinaten X und Z handelt es sich hier wieder um ein Randwertproblem und bezogen auf die Zeitkoordinate T um ein Anfangswertproblem.

Das Differenzenverfahren ist nur für lineare Gleichungen geeignet. Um das bewährte Verfahren auch hier anwenden zu können, wird deshalb die nicht lineare Differentialgleichung durch eine linearisierte Näherung ersetzt. Ausgehend von geschätzten **Anfangsnäherungswerten Pn** für den Druck P kann so über mehrere Iterationsschritte auch dieses Problem mit Hilfe des Differenzenverfahrens gelöst werden. Dabei wird in folgender Weise vorgegangen: Zunächst wird durch Extrapolation aus der Druckverteilung im Schmierspalt des vorhergehenden Zeitpunktes J_T-1 eine erste Näherung Pn für den Zeitpunkt J_T geschätzt. Anschließend wird über NI Iterationszyklen durch Anwendung des Differenzenverfahrens auf die linearisierte Differentialgleichung die Näherungslösung verbessert. Es hat sich gezeigt, dass üblicherweise nach der Extrapolation 3 Iterationszyklen ausreichend sind. Deshalb ist im Programm NI=3 vorgegeben. Der Parameter NI kann im Eingabemenü 4.4.12.3 "Programminterne Parameter bearbeiten" verändert werden, wovon aber in der Regel abzuraten ist. Dieser Iterationszyklus ist in den Bildern 3.04, und 3.06 als grüne Schleife dargestellt. Etwas detailierter ist er außerdem in der Übersichtsdarstellung des Flussdiagramms der Routine "FilmDruck2" (Bild 3.10 rechtes Diagramm) dargestellt.

Durch die Linearisierung wird die erweiterte Reynoldssche Differentialgleichung in die lineare Form gebracht

$$(3.020)\qquad \frac{\partial^2 P}{\partial X^2} + A_1\cdot\frac{\partial^2 P}{\partial Z^2} + A_2\cdot\frac{\partial P}{\partial X} + A_3\cdot\frac{\partial P}{\partial Z} + A_4\cdot P = R$$

wobei

$$(3.021)\qquad C_1 = 1/B^2$$

$$(3.022)\qquad C_2 = \Omega/2$$

$$(3.023)\qquad C_3 = 6/\pi$$

$$(3.024)\qquad C_4 = \frac{1}{P(T-\Delta T)\cdot\Delta T}$$

$$(3.025)\qquad \frac{\partial Pn}{\partial T} = C_4\cdot Pn\cdot(Pn - P(T-\Delta T))$$

$$(3.026)\qquad C_5 = 3/H$$

$$(3.027)\qquad C_6 = C_3/H^2$$

$$(3.028)\qquad C_7 = \frac{1}{Pn+C}$$

$$(3.029)\qquad C_8 = C_7^2$$

$$(3.030)\qquad C_{10} = C_2\cdot\frac{\partial H}{\partial X} + \frac{\partial H}{\partial T}$$

$$(3.031)\qquad C_{11} = C_2\cdot\frac{\partial Pn}{\partial X} + \frac{\partial Pn}{\partial T}$$

$$(3.032)\qquad C_{12} = C_6\cdot C_8\cdot C$$

$$(3.033)\qquad C_{13} = C_7\cdot 2\cdot C_{11}$$

$$(3.034)\qquad A_1 = C_1$$

$$(3.035)\qquad A_2 = C_5\cdot\frac{\partial H}{\partial X} - C_{12}\cdot C_2$$

$$(3.036)\qquad A_3 = C_1\cdot C_5\cdot\frac{\partial H}{\partial Z}$$

$$A_4 = C_{12} \cdot \left[-\frac{C_{10}}{H} + C_{13} - C_4 \cdot \{2 \cdot Pn - P(T - \Delta T)\} \right] \quad (3.037)$$

$$R = C_{12} \cdot Pn \cdot \left(\frac{C_{10} \cdot Pn}{H \cdot C} + C_{13} - C_4 \cdot Pn \right) \quad (3.038)$$

Pn ist hier die vorhergehende Näherung für den Druck P. Ausführlich erläutert ist die Linearisierung in [20, Abschnitt 7.3].

Die Felder der Koeffizienten $A_1(N_{Glei})$ bis $A_4(N_{Glei})$ und die rechten Seiten $R(N_{Glei})$ für jedes Flächenelement (J_Z,J_X) des eigentlichen Schmierspalts werden durch die Routine "KeoDGL7" berechnet.

HINWEIS: *Die Flächen der Schmiertaschen zählen nicht zum eigentlichen Schmierspalt.*

Die iterative Berechnung des Druckverlaufs im Schmierspalt mit der linearisierten erweiterten Reynoldsschen Differentialgleichung wird durch die Routine "FilmDruck2" ausgeführt.

3.4.1.4 Diskretisierung der Schmierspaltfläche

Um das Differenzenverfahren anwenden zu können, muss die kontinuierliche Funktion der Druckverteilung im Schmierspalt diskretisiert werden. Gemäß Bild 3.19 wird die Fläche des abgewickelten Schmierspalts in $N_X \cdot N_Z$ Flächenelemente der Größe $\Delta X \cdot \Delta Z$ aufgeteilt. Im gezeigten Beispiel handelt es sich um ein voll umschlossenes (Vollum=1), symmetrisches (Sym=1) Lager.

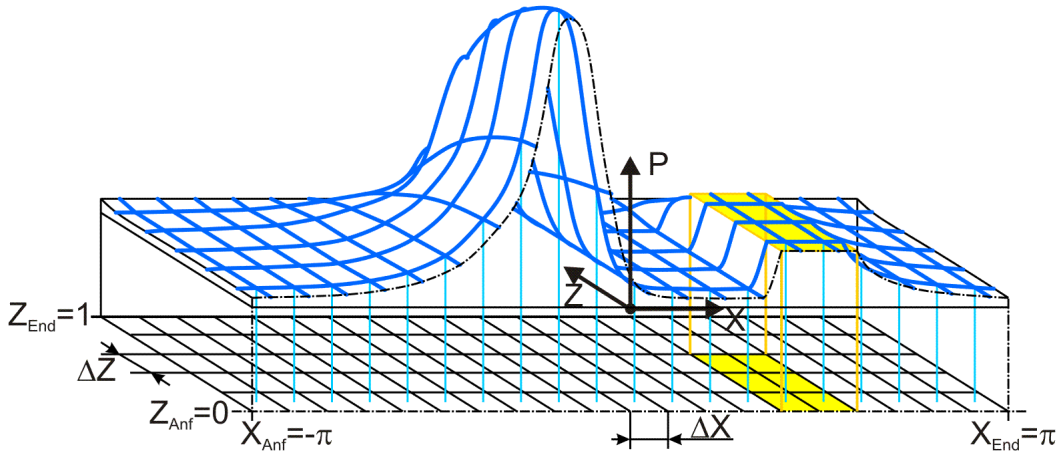


Bild 3.19: *Diskretisierung der Schmierspaltfläche in $N_X \cdot N_Z$ Flächenelemente*

Die Drücke $P(J_Z,J_X)$ werden jeweils für die Mitte des Flächenelements (J_Z,J_X) berechnet (**hellblaue** Stützstellen). Mit dem **dunkelblauen** Liniennetz ist hier die Druckverteilung im Schmierspalt skizziert. Der Druckverlauf zwischen den berechneten Stützstellen wird durch Parabelbögen approximiert.

Die gelbe Fläche im Bild 3.19 zeigt die Ausdehnung einer Schmiertasche. Die freie Wahl der Anordnung von Schmiertaschen ist lediglich dadurch eingeschränkt, dass der Verlauf der Taschenränder immer entlang der Grenzen der Flächenelemente angenommen wird. Innerhalb der gesamten Taschenfläche wird ein konstanter Druck angenommen. Das wird durch entsprechende Approximations-gleichungen berücksichtigt. Siehe dazu Abschnitt 3.4.1.6.

Mit dem Grafikprogramm GNUPLOT können die berechneten Druckverteilungen im Schmierspalt schnell dargestellt werden, wobei auf die Gitterteilung der Berechnung zurückgegriffen wird. Es ist dabei jedoch zu beachten, dass hier die Darstellung, bedingt durch die Eigenheiten des Programms GNUPLOT, etwas vereinfacht und teilweise eingeschränkt ist. Bild 3.20 zeigt den Druckverlauf für das Beispiel aus Bild 3.19 in der Darstellung mit GNUPLOT.

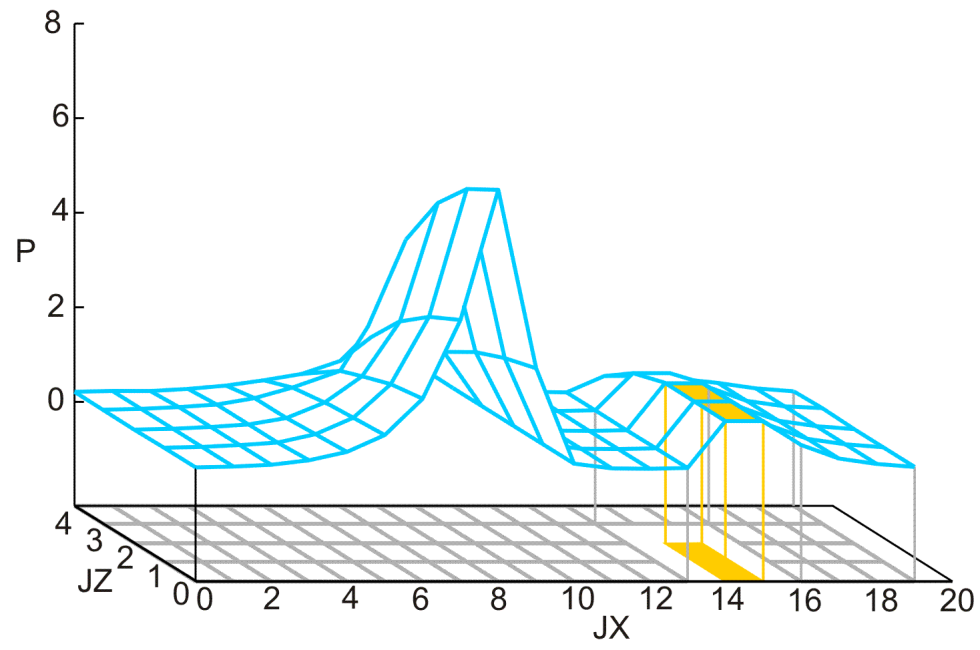


Bild 3.20: *Darstellung der Druckverteilung im Schmierspalt mit dem Grafikprogramm GNUPLOT*

Hier werden lediglich die berechneten Drücke in den Mitten der Flächenelemente angezeigt, verbunden durch gerade Linien zu einem Gitternetz. Dadurch wird die Druckverteilung nicht exakt bis an den Lagerrand dargestellt. Es fehlt rundherum jeweils die Breite eines halben Flächenelements und die Zählung der Gitterpunkte beginnt nicht wie in der Berechnung mit 1 sondern mit 0. Das in GNUPLOT gezeigte Gitternetz (**grau**) zeigt nicht die Flächenelemente, die der Berechnung zugrundeliegen, sondern die Verbindungslinien der Elementmitten. Die berechneten Druckwerte liegen deshalb hier über den Schnittpunkten des GNUPLOT-Gitternetzes.

Insbesondere die Darstellung der Schmiertaschen kann durch diese vereinfachte Darstellung bei einem kritischen Beobachter evtl. zu Irritationen führen. Während in der Darstellung der Druckverteilung die Schmiertasche als Fläche mit einem konstanten Druck (orange Fläche) um jeweils eine Flächenelementbreite zu klein dargestellt wird, erscheint die Schmiertasche in dem darunter liegenden Gitternetz um eine Flächenelementbreite zu groß.

Die Überlagerung der beiden oben gezeigten Bilder im Bild 3.21 soll diese Unterschiede noch einmal verdeutlichen.

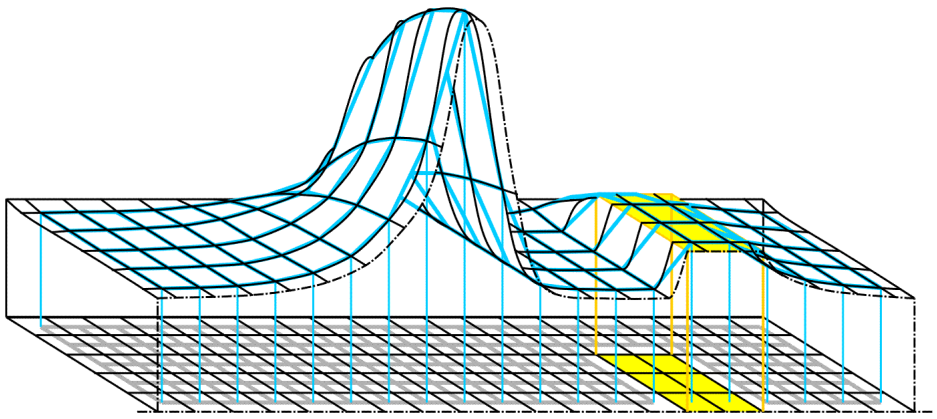


Bild 3.21: *Überlagerung der grafischen Darstellungen aus Bild 3.19 und 3.20*

In dieser Darstellung stellen die gelben Flächen wieder die wahre Größe der Schmiertasche dar und die genauere Darstellung der Druckverteilung aus Bild 3.19 ist hier mit dünnen schwarzen Linien dargestellt.

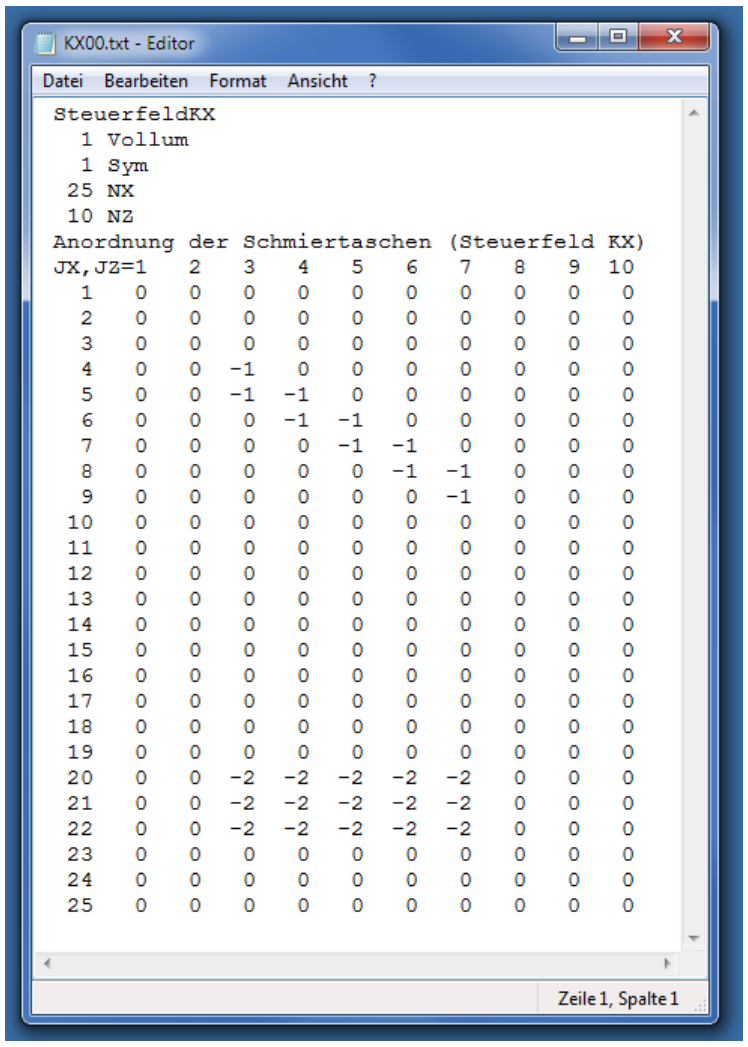
Bei der hier gewählten groben Gitterteilung von 20·5 Flächenelementen werden die Abweichung der mit GNUPLOT dargestellten Druckverteilung gegenüber der im Modell approximierten Druckverteilung deutlich. Je feiner jedoch die Gitterteilung gewählt wird, umso unbedeutender werden diese Abweichungen. Deshalb wurde auch darauf verzichtet, weiteren Aufwand in eine verbesserte Darstellung zu investieren.

Die Eingabe- und Ergebnisdaten zu dem hier gezeigten Beispiel sind im Demonstrationsbeispiel "Demo22 " abgelegt.

3.4.1.5 Darstellung von Schmiertaschen im Schmierspalt

Nur wenn sich innerhalb des Schmierspaltes keine Schmiertaschen befinden und das Lager ausschließlich über den Lagerrand geschmiert wird, ist für alle $N_X \cdot N_Z$ Flächenelemente die Differenzengleichung aufzustellen. Falls im Lager Schmiertaschen angeordnet sind, sind deren Flächen aus dem eigentlichen Schmierspalt auszuschließen. Weil in den Schmiertaschen die Spalthöhe viel größer ist als im eigentlichen Schmierspalt, wird angenommen, dass der Druck über die Fläche einer Schmiertasche konstant ist und hier die Reynoldssche Gleichung deshalb nicht zur Anwendung kommt. Allerdings sind auch für die Berechnung der Drücke in den Schmiertaschen einige Gleichungen aufzustellen, was im Abschnitt 3.4.2 beschrieben wird.

Um die Lage und Ausdehnung der Schmiertaschen innerhalb der abgewickelten Spaltfläche zu erfassen, wurde das Steuerfeld KX eingerichtet, welches durch ein Feld von $N_Z \cdot N_X$ ganzen Zahlen die Anordnung der diskretisierten Flächenelemente abbildet. Alle Flächenelemente, die Teil einer Schmiertasche sind, werden mit einer negativen ganzen Zahl im Steuerfeld KX belegt, deren Betrag die Nummer der jeweiligen Schmiertasche codiert. Bild 4.020 zeigt das Steuerfeld KX eines symmetrischen, voll umschlossenen Lagers, dessen Spaltfläche in 25·10 Flächenelemente aufgeteilt wurde, mit 2 Schmiertaschen.



KX00.txt - Editor

Datei Bearbeiten Format Ansicht ?

```
 SteuerfeldKX
   1 Vollum
   1 Sym
  25 NX
  10 NZ
Anordnung der Schmiertaschen (Steuerfeld KX)
JX,JZ=1    2    3    4    5    6    7    8    9   10
   1    0    0    0    0    0    0    0    0    0    0
   2    0    0    0    0    0    0    0    0    0    0
   3    0    0    0    0    0    0    0    0    0    0
   4    0    0   -1    0    0    0    0    0    0    0
   5    0    0   -1   -1    0    0    0    0    0    0
   6    0    0    0   -1   -1    0    0    0    0    0
   7    0    0    0    0   -1   -1    0    0    0    0
   8    0    0    0    0    0   -1   -1    0    0    0
   9    0    0    0    0    0    0   -1    0    0    0
  10    0    0    0    0    0    0    0    0    0    0
  11    0    0    0    0    0    0    0    0    0    0
  12    0    0    0    0    0    0    0    0    0    0
  13    0    0    0    0    0    0    0    0    0    0
  14    0    0    0    0    0    0    0    0    0    0
  15    0    0    0    0    0    0    0    0    0    0
  16    0    0    0    0    0    0    0    0    0    0
  17    0    0    0    0    0    0    0    0    0    0
  18    0    0    0    0    0    0    0    0    0    0
  19    0    0    0    0    0    0    0    0    0    0
  20    0    0   -2   -2   -2   -2   -2    0    0    0
  21    0    0   -2   -2   -2   -2   -2    0    0    0
  22    0    0   -2   -2   -2   -2   -2    0    0    0
  23    0    0    0    0    0    0    0    0    0    0
  24    0    0    0    0    0    0    0    0    0    0
  25    0    0    0    0    0    0    0    0    0    0
```

Zeile 1, Spalte 1

Bild 4.020: *Beispiel eines Steuerfeldes KX mit N_X=25 Zeilen, N_Z=10 Spalten und 2 Schmiertaschen für ein voll umschlossenes, symmetrisches Lager, wie es in einer Textdatei abgespeichert ist*

Anhand dieses Feldes kann das Programm erkennen, für welche Feldelemente keine Differenzengleichungen aufzustellen sind und welche anderen Gleichungen dafür evtl. in das Gleichungssystem einzufügen sind. Zur Eingabe von Schmiertaschen siehe in der Bedienanleitung den Abschnitt 4.4.8.

HINWEIS: *Die Anordnung von Schmiertaschen ist im Programm SIRIUS nur für die Lagerschale vorgesehen. Das ist das Teil des Lagers, mit dem die Koordinatensysteme x-y-z und 1-2-3 fest verbunden sind. Es können aber auch die Schmiertaschen in der Welle modelliert werden. Dazu wird die Welle einfach als Lagerschale im Programm SIRIUS angenommen. Zu beachten ist hier, dass die Koordinatensysteme x-y-z und 1-2-3 mit der Welle mitbewegt werden und dass dem die Darstellung aller anderen Eingabeparameter anzupassen ist.*

Schmiertaschen sowohl auf der Lagerschale als auch auf der Welle anzuordnen, ist im Programm nicht vorgesehen. Davon gibt es eine Ausnahme: Neben beliebig geformten und angeordneten Schmiertaschen auf einer Gleitfläche können auch Ringnuten, die das Lager vollständig umschließen, auf der anderen Gleitfläche angeordnet sein. Das ist möglich, weil es bei Ringnuten egal ist, auf welcher der

beiden Gleitflächen des Lagers sie angeordnet sind. Ihre Wirkung auf die Schmiermittelströmung im Schmierspalt ist die gleiche. Deshalb können z.B. Ringnuten in der Wellenoberfläche einfach auf der Lagerschalenoberfläche modelliert werden.

3.4.1.6 Liste der verwendeten Differenzengleichungen und ihre Codierung in den Steuerfeldern KX, KZ und NG

Bild 3.23 zeigt zwei willkürliche Anordnungen von Schmiertaschen, links in einem voll umschlossenen, symmetrischen Lager und rechts in einem teilweise umschlossenen, asymmetrischen Lager. Anhand dieser zwei Beispiele soll die Verwendung aller erforderlichen Varianten von Differenzengleichungen gezeigt werden, die aus den Gleichungen (3.010 bis 3.015) der 3 Basisvarianten (Abschnitt 3.4.1.1) abgeleitet sind.

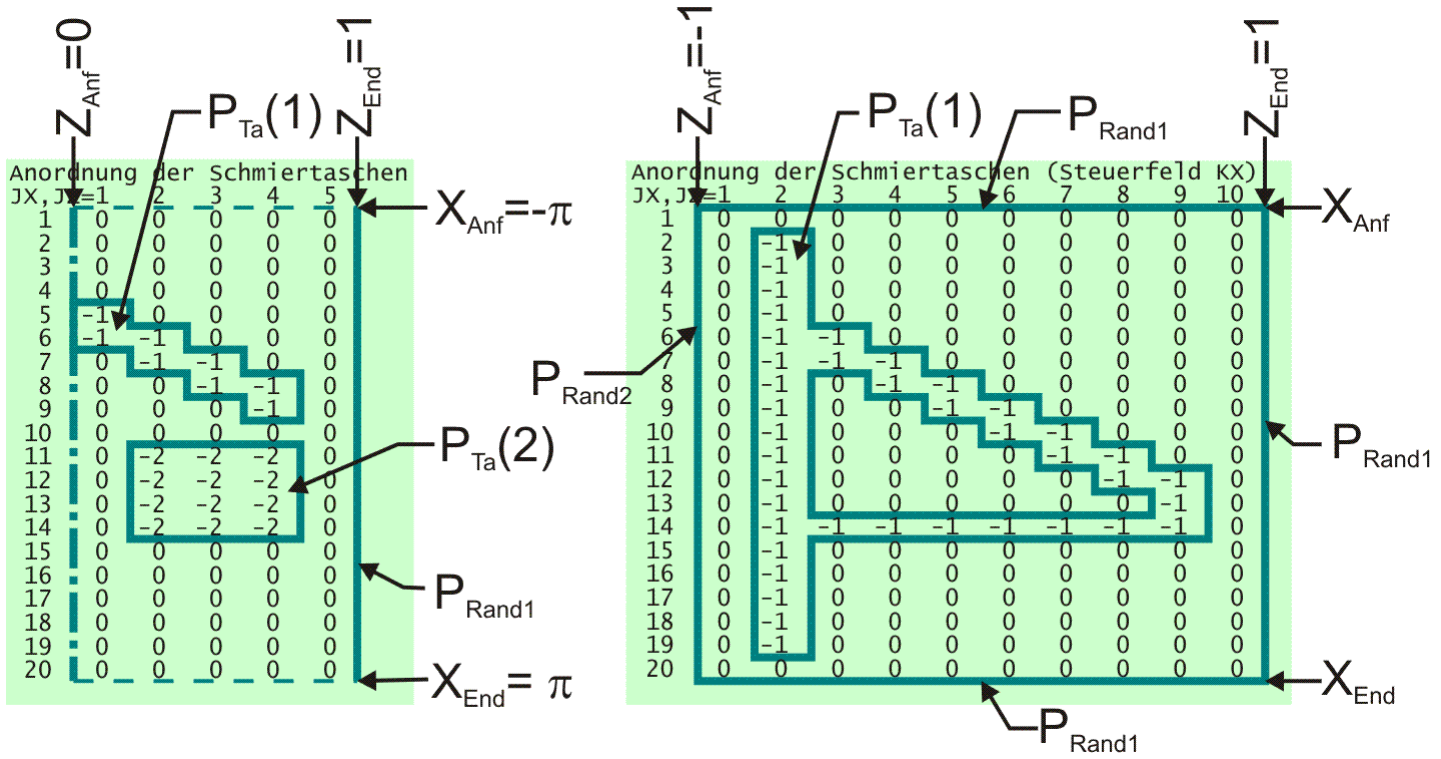


Bild 3.23: *Zwei Varianten von Schmiertaschenanordnungen*

Aus der Anordnung der Schmiertaschen ist ersichtlich, dass in Abhängigkeit von der Lage eines Flächenelements verschiedene Differenzengleichungen verwendet werden müssen. Für die Berechnung der partiellen Ableitungen in X-Richtung ergeben sich dabei 8 Varianten und für die Ableitungen in Z-Richtung ergeben sich 10 Varianten, die nachfolgend angegeben werden:

Variante 1: Das Flächenelement (J_Z,J_X) des Schmierspalts grenzt in Richtung der entsprechenden Koordinatenachse an beiden Seiten an Flächenelemente des eigentlichen Schmierspalts. Bild 3.24 skizziert diesen Fall für die X-Richtung.

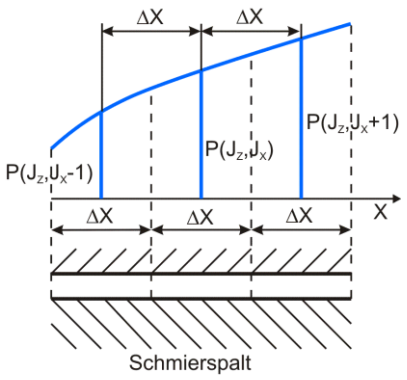


Bild 3.24: *Skizze zu Variante 1*

Für die partiellen Ableitungen in X-Richtung gilt

$$\frac{\partial P(J_Z,J_X)}{\partial X} \approx \frac{-3\cdot P(J_Z,J_X-1)+3\cdot P(J_Z,J_X+1)}{6\cdot \Delta X} \qquad (3.050)$$

$$\frac{\partial^2 P(J_Z,J_X)}{\partial X^2} \approx \frac{6\cdot P(J_Z,J_X-1)-12\cdot P(J_Z,J_X)+6\cdot P(J_Z,J_X+1)}{6\cdot \Delta X^2} \qquad (3.051)$$

Für den analogen Fall in Z-Richtung gilt

$$\frac{\partial P(J_Z,J_X)}{\partial Z} \approx \frac{-3\cdot P(J_Z-1,J_X)+3\cdot P(J_Z+1,J_X)}{6\cdot \Delta Z} \qquad (3.052)$$

$$\frac{\partial^2 P(J_Z,J_X)}{\partial Z^2} \approx \frac{6\cdot P(J_Z-1,J_X)-12\cdot P(J_Z,J_X)+6\cdot P(J_Z+1,J_X)}{6\cdot \Delta Z^2} \qquad (3.053)$$

Variante 2: Das Flächenelement (J_Z,J_X) des Schmierspalts grenzt in negativer Richtung der Koordinatenachse an ein Flächenelement, das zu einer Schmiertasche gehört. Es wird angenommen, dass die Grenze zwischen Schmiertasche und Schmierspaltelement genau auf der Grenze zwischen den Flächenelementen liegt und im gesamten Flächenelement der Schmiertasche ein konstanter Druck herrscht. Bild 3.25 skizziert diesen Fall für die X-Richtung.

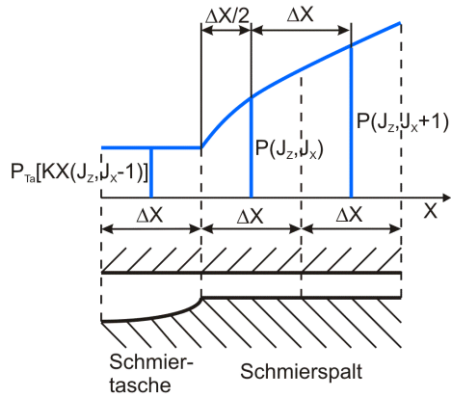


Bild 3.25: *Skizze zu Variante 2*

Für die partiellen Ableitungen in X-Richtung gilt

$$\frac{\partial P(J_Z,J_X)}{\partial X} \approx \frac{-8\cdot P_{Ta}\left[-KX(J_Z,J_X-1)\right]+6\cdot P(J_Z,J_X)+2\cdot P(J_Z,J_X+1)}{6\cdot \Delta X} \qquad (3.054)$$

$$\frac{\partial^2 P(J_Z,J_X)}{\partial X^2} \approx \frac{16\cdot P_{Ta}\left[-KX(J_Z,J_X-1)\right]-24\cdot P(J_Z,J_X)+8\cdot P(J_Z,J_X+1}{6\cdot \Delta X^2} \qquad (3.055)$$

Für den analogen Fall in Z-Richtung gilt

$$\frac{\partial P(J_Z,J_X)}{\partial Z} \approx \frac{-8\cdot P_{Ta}\left[-KX(J_Z-1,J_X)\right]+6\cdot P(J_Z,J_X)+2\cdot P(J_Z+1,J_X)}{6\cdot \Delta Z} \qquad (3.056)$$

$$\frac{\partial^2 P(J_Z,J_X)}{\partial Z^2} \approx \frac{16\cdot P_{Ta}\left[-KX(J_Z-1,J_X)\right]-24\cdot P(J_Z,J_X)+8\cdot P(J_Z+1,J_X}{6\cdot \Delta Z^2} \qquad (3.057)$$

Variante 3: Das Flächenelement (J_Z,J_X) des Schmierspalts grenzt in positiver Richtung der Koordinatenachse an ein Flächenelement, das zu einer Schmiertasche gehört. Es wird angenommen, dass die Grenze zwischen Schmiertasche und Schmierspaltelement genau auf der

Grenze zwischen den Flächenelementen liegt und im gesamten Flächenelement der Schmiertasche ein konstanter Druck herrscht. Bild 3.26 skizziert diesen Fall für die X-Richtung.

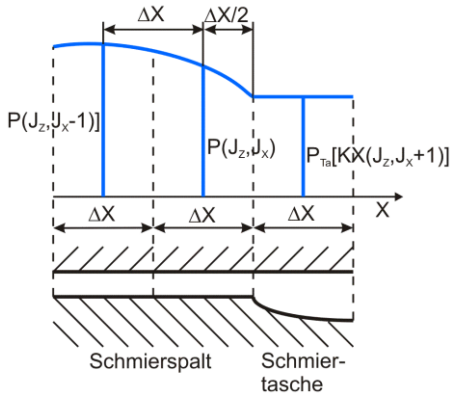


Bild 3.26: Skizze zu Variante 3

Für die partiellen Ableitungen in X-Richtung gilt

$$(3.058)\qquad \frac{\partial P(J_Z,J_X)}{\partial X} \approx \frac{-2\cdot P(J_Z,J_X-1)-6\cdot P(J_Z,J_X)+8\cdot P_{Ta}\left[-KX(J_Z,J_X+1)\right]}{6\cdot \Delta X}$$

$$(3.059)\qquad \frac{\partial^2 P(J_Z,J_X)}{\partial X^2} \approx \frac{8\cdot P(J_Z,J_X-1)-24\cdot P(J_Z,J_X)+16\cdot P_{Ta}\left[-KX(J_Z,J_X+1)\right]}{6\cdot \Delta X^2}$$

Für den analogen Fall in Z-Richtung gilt

$$(3.060)\qquad \frac{\partial P(J_Z,J_X)}{\partial Z} \approx \frac{-2\cdot P(J_Z-1,J_X)-6\cdot P(J_Z,J_X)+8\cdot P_{Ta}\left[-KX(J_Z+1,J_X)\right]}{6\cdot \Delta Z}$$

$$(3.061)\qquad \frac{\partial^2 P(J_Z,J_X)}{\partial Z^2} \approx \frac{8\cdot P(J_Z-1,J_X)-24\cdot P(J_Z,J_X)+16\cdot P_{Ta}\left[-KX(J_Z+1,J_X)\right]}{6\cdot \Delta Z^2}$$

Variante 4: Das Flächenelement (J_Z,J_X) des Schmierspalts grenzt in Richtung der entsprechenden Koordinatenachse an beiden Seiten an Flächenelemente von Schmiertaschen. Bild 3.27 skizziert diesen Fall für die X-Richtung.

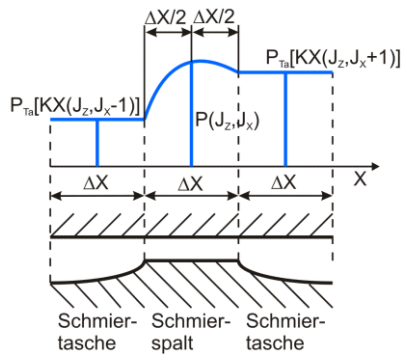


Bild 3.27: Skizze zu Variante 4

Für die partiellen Ableitungen in X-Richtung gilt

$$(3.062)\qquad \frac{\partial P(J_Z,J_X)}{\partial X} \approx \frac{-6\cdot P_{Ta}\left[-KX(J_Z,J_X-1)\right]+6\cdot P_{Ta}\left[-KX(J_Z,J_X+1)\right]}{6\cdot \Delta X}$$

$$(3.063)\qquad \frac{\partial^2 P(J_Z,J_X)}{\partial X^2} \approx \frac{24\cdot P_{Ta}\left[-KX(J_Z,J_X-1)\right]-48\cdot P(J_Z,J_X)+24\cdot P_{Ta}\left[-KX(J_Z,J_X+1)\right]}{6\cdot \Delta X^2}$$

Für den analogen Fall in Z-Richtung gilt

$$(3.064)\qquad \frac{\partial P(J_Z,J_X)}{\partial Z} \approx \frac{-6\cdot P_{Ta}\left[-KX(J_Z-1,J_X)\right]+6\cdot P_{Ta}\left[-KX(J_Z+1,J_X)\right]}{6\cdot \Delta Z}$$

$$(3.065)\qquad \frac{\partial^2 P(J_Z,J_X)}{\partial Z^2} \approx \frac{24\cdot P_{Ta}\left[-KX(J_Z-1,J_X)\right]-48\cdot P(J_Z,J_X)+24\cdot P_{Ta}\left[-KX(J_Z+1,J_X)\right]}{6\cdot \Delta Z^2}$$

Variante 5: Das Flächenelement (J_Z,J_X) des Schmierspalts grenzt in negativer Richtung der Koordinatenachse an den Rand des Lagers. Bild 3.28 skizziert diesen Fall für die X-Richtung.

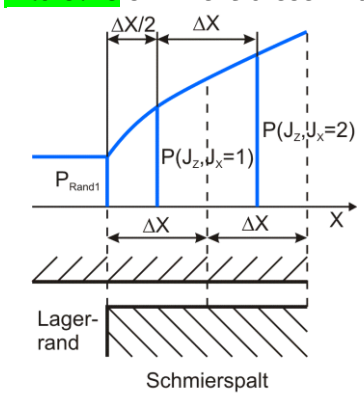


Bild 3.28: Skizze zu Variante 5

Für die partiellen Ableitungen in X-Richtung gilt

$$(3.066)\qquad \frac{\partial P(J_Z,J_X=1)}{\partial X} \approx \frac{-8\cdot P_{Rand1}+6\cdot P(J_Z,J_X=1)+2\cdot P(J_Z,J_X=2)}{6\cdot \Delta X}$$

$$(3.067)\qquad \frac{\partial^2 P(J_Z,J_X=1)}{\partial X^2} \approx \frac{16\cdot P_{Rand1}-24\cdot P(J_Z,J_X=1)+8\cdot P(J_Z,J_X=2)}{6\cdot \Delta X^2}$$

Für den analogen Fall in Z-Richtung gilt

$$(3.068)\qquad \frac{\partial P(J_Z=1,J_X)}{\partial Z} \approx \frac{-8\cdot P_{Rand2}+6\cdot P(J_Z=1,J_X)+2\cdot P(J_Z=2,J_X)}{6\cdot \Delta Z}$$

$$(3.069)\qquad \frac{\partial^2 P(J_Z1,J_X)}{\partial Z^2} \approx \frac{16\cdot P_{Rand2}-24\cdot P(J_Z=1,J_X)+8\cdot P(J_Z=2,J_X)}{6\cdot \Delta Z^2}$$

Variante 6: Das Flächenelement (J_Z,J_X) des Schmierspalts grenzt in der negativen Richtung der entsprechenden Koordinatenachse an den Lagerrand und in positive Richtung an das Flächenelement einer Schmiertasche. Bild 3.29 skizziert diesen Fall für die X-Richtung.

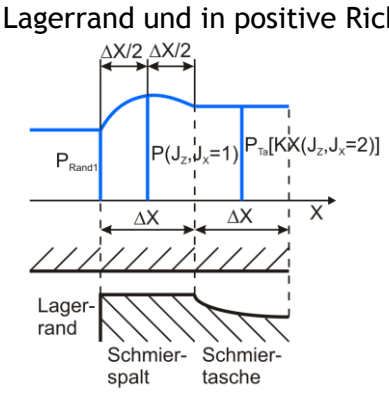


Bild 3.29: Skizze zu Variante 6

Für die partiellen Ableitungen in X-Richtung gilt

$$(3.070)\qquad \frac{\partial P(J_Z,J_X=1)}{\partial X} \approx \frac{-6\cdot P_{Rand1}+6\cdot P_{Ta}\left[-KX(J_Z,J_X=2)\right]}{6\cdot \Delta X}$$

$$(3.071)\qquad \frac{\partial^2 P(J_Z,J_X=1)}{\partial X^2} \approx \frac{24\cdot P_{Rand1}-48\cdot P(J_Z,J_X=1)+24\cdot P_{Ta}\left[-KX(J_Z,J_X=2)\right]}{6\cdot \Delta X^2}$$

Für den analogen Fall in Z-Richtung gilt

$$(3.072)\qquad \frac{\partial P(J_Z=1,J_X)}{\partial Z} \approx \frac{-6\cdot P_{Rand2}+6\cdot P_{Ta}\left[-KX(J_Z=2,J_X)\right]}{6\cdot \Delta Z}$$

$$(3.073)\qquad \frac{\partial^2 P(J_Z=1,J_X)}{\partial Z^2} \approx \frac{24\cdot P_{Rand2}-48\cdot P(J_Z=1,J_X)+24\cdot P_{Ta}\left[-KX(J_Z=2,J_X)\right]}{6\cdot \Delta Z^2}$$

Variante 7: Das Flächenelement (J_Z,J_X) des Schmierspalts grenzt in positiver Richtung der Koordinatenachse an den Lagerrand. Bild 3.30 skizziert diesen Fall für die X-Richtung.

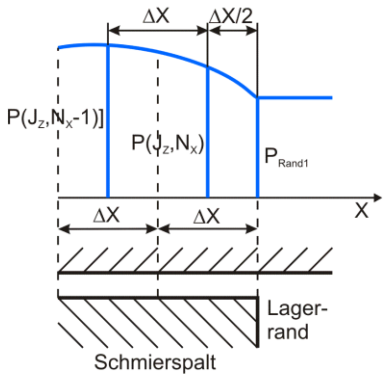


***Bild 3.30**: Skizze zu Variante 7*

Für die partiellen Ableitungen in X-Richtung gilt

$$(3.074)\qquad \frac{\partial P(J_Z,N_X)}{\partial X} \approx \frac{-2\cdot P(J_Z,N_X-1)-6\cdot P(J_Z,N_X)+8\cdot P_{Rand1}}{6\cdot \Delta X}$$

$$(3.075)\qquad \frac{\partial^2 P(J_Z,N_X)}{\partial X^2} \approx \frac{8\cdot P(J_Z,N_X-1)-24\cdot P(J_Z,N_X)+16\cdot P_{Rand1}}{6\cdot \Delta X^2}$$

Für den analogen Fall in Z-Richtung gilt

$$(3.076)\qquad \frac{\partial P(N_Z,J_X)}{\partial Z} \approx \frac{-2\cdot P(N_Z-1,J_X)-6\cdot P(N_Z,J_X)+8\cdot P_{Rand1}}{6\cdot \Delta Z}$$

$$(3.077)\qquad \frac{\partial^2 P(N_Z,J_X)}{\partial Z^2} \approx \frac{8\cdot P(N_Z-1,J_X)-24\cdot P(N_Z,J_X)+16\cdot P_{Rand1}}{6\cdot \Delta Z^2}$$

Variante 8: Das Flächenelement (J_Z,J_X) des Schmierspalts grenzt in negativer Richtung der entsprechenden Koordinatenachse an das Flächenelement einer Schmiertasche und in positiver Richtung an den Lagerrand. Bild 3.31 skizziert diesen Fall für die X-Richtung.

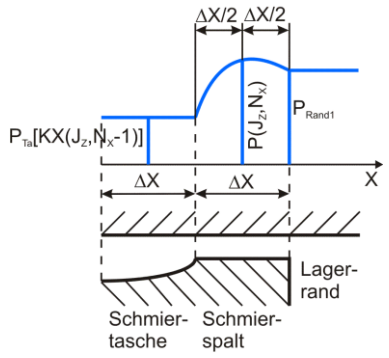


***Bild 3.31**: Skizze zu Variante 8*

Für die partiellen Ableitungen in X-Richtung gilt

$$(3.078)\qquad \frac{\partial P(J_Z,N_X)}{\partial X} \approx \frac{-6\cdot P_{Ta}\left[-KX(J_Z,N_X-1)\right]+6\cdot P_{Rand1}}{6\cdot \Delta X}$$

$$(3.079)\qquad \frac{\partial^2 P(J_Z,N_X)}{\partial X^2} \approx \frac{24\cdot P_{Ta}\left[-KX(J_Z,N_X-1)\right]-48\cdot P(J_Z,N_X)+24\cdot P_{Rand1}}{6\cdot \Delta X^2}$$

Für den analogen Fall in Z-Richtung gilt

$$(3.080)\qquad \frac{\partial P(N_Z,J_X)}{\partial Z} \approx \frac{-6\cdot P_{Ta}\left[-KX(N_Z-1,J_X)\right]+6\cdot P_{Rand1}}{6\cdot \Delta Z}$$

$$(3.081)\qquad \frac{\partial^2 P(N_Z,J_X)}{\partial Z^2} \approx \frac{24\cdot P_{Ta}\left[-KX(N_Z-1,J_X)\right]-48\cdot P(N_Z,J_X)+24\cdot P_{Rand1}}{6\cdot \Delta Z^2}$$

Für die Berechnung der Differenzenquotienten $\partial P/\partial Z$ und $\partial^2 P/\partial Z^2$ gibt es wegen der möglichen Symmetrie im Lager noch 2 weitere Varianten.

Variante 9: Das Flächenelement (J_Z,J_X) des Schmierspalts grenzt in Richtung der negativen Z-Achse an die Symmetrieebene des Lagers und in Richtung der positiven Z-Achse an ein Flächenelement des eigentlichen Schmierspalts. Bild 3.32 skizziert diesen Fall für die Z-Richtung.

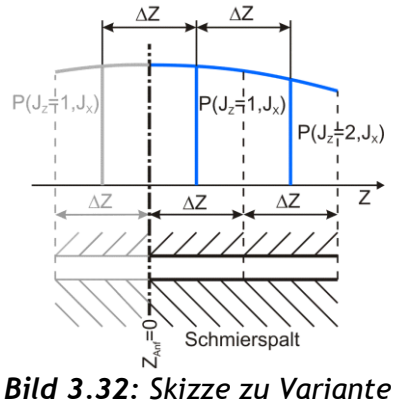


***Bild 3.32**: Skizze zu Variante 9*

Für die partiellen Ableitungen in Z-Richtung gilt

$$(3.082)\qquad \frac{\partial P(J_Z=1,J_X)}{\partial Z} \approx \frac{-3\cdot P(J_Z-1,J_X)+3\cdot P(J_Z=2,J_X)}{6\cdot \Delta Z}$$

$$(3.083)\qquad \frac{\partial^2 P(J_Z=1,J_X)}{\partial Z^2} \approx \frac{-6\cdot P(J_Z=1,J_X)+6\cdot P(J_Z=2,J_X)}{6\cdot \Delta Z^2}$$

Variante 10: Das Flächenelement (J_Z,J_X) des Schmierspalts grenzt in Richtung der negativen Z-Achse an die Symmetrieebene des Lagers und in Richtung der positiven Z-Achse an das Flächenelement einer Schmiertasche. Bild 3.33 skizziert diesen Fall für die Z-Richtung.

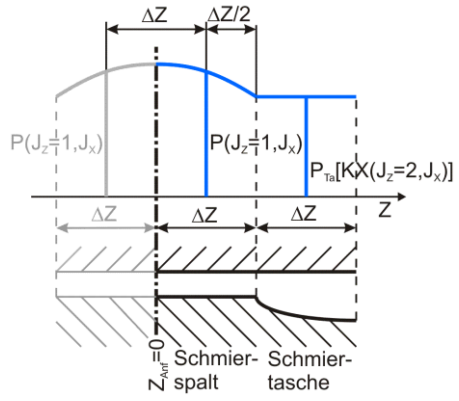


***Bild 3.33**: Skizze zu Variante 10*

Für die partiellen Ableitungen in Z-Richtung gilt

$$(3.084)\qquad \frac{\partial P(J_Z=1,J_X)}{\partial Z} \approx \frac{-8\cdot P(J_Z=1,J_X)+8\cdot P_{Ta}\left[-KX(J_Z=2,J_X)\right]}{6\cdot \Delta Z}$$

$$(3.085)\qquad \frac{\partial^2 P(J_Z=1,J_X)}{\partial Z^2} \approx \frac{-16\cdot P(J_Z=1,J_X)+16\cdot P_{Ta}\left[-KX(J_Z=2,J_X)\right]}{6\cdot \Delta Z^2}$$

Welche Formeln auf welches Flächenelement angewendet werden müssen, wird durch positive ganzzahlige Werte in den Steuerfeldern KX und KZ codiert abgelegt. Im Steuerfeld KX werden programmintern neben der Anordnung der Schmiertaschen auch die Variantennummern der entsprechenden Differenzenformeln für die partiellen Ableitungen $\partial P/\partial X$ und $\partial^2 P/\partial X^2$ abgelegt. Bild 3.34 zeigt die Steuerfelder KX für die 2 Lagervarianten analog zu Bild 3.23, wie sie programmintern mit diesen Variantennummern aufgefüllt wurden. Die Nummerierung der Varianten im Feld KX ist identisch mit der Nummerierung der Varianten 1 bis 8 hier in der Dokumentation.

Feld der Punkttypen KX

JX,JZ=1	2	3	4	5
1	1	1	1	1
2	1	1	1	1
3	1	1	1	1
4	3	1	1	1
5	-1	3	1	1
6	-1	-1	3	1
7	2	-1	-1	3
8	1	2	-1	-1
9	1	1	2	-1
10	1	3	3	4
11	1	-2	-2	-2
12	1	-2	-2	-2
13	1	-2	-2	-2
14	1	-2	-2	-2
15	1	2	2	2
16	1	1	1	1
17	1	1	1	1
18	1	1	1	1
19	1	1	1	1
20	1	1	1	1

Feld der Punkttypen KX

JX,JZ=1	2	3	4	5	6	7	8	9	10
1	5	6	5	5	5	5	5	5	5
2	1	-1	1	1	1	1	1	1	1
3	1	-1	1	1	1	1	1	1	1
4	1	-1	1	1	1	1	1	1	1
5	1	-1	3	1	1	1	1	1	1
6	1	-1	-1	3	1	1	1	1	1
7	1	-1	-1	-1	3	1	1	1	1
8	1	-1	2	-1	-1	3	1	1	1
9	1	-1	1	2	-1	-1	3	1	1
10	1	-1	1	1	2	-1	-1	3	1
11	1	-1	1	1	1	2	-1	-1	3
12	1	-1	1	1	1	1	2	-1	-1
13	1	-1	3	3	3	3	3	4	-1
14	1	-1	-1	-1	-1	-1	-1	-1	-1
15	1	-1	2	2	2	2	2	2	2
16	1	-1	1	1	1	1	1	1	1
17	1	-1	1	1	1	1	1	1	1
18	1	-1	1	1	1	1	1	1	1
19	1	-1	1	1	1	1	1	1	1
20	7	8	7	7	7	7	7	7	7

Bild 3.34: *Darstellung der vollständigen Steuerfelder KX für die 2 Varianten von Schmiertaschenanordnungen analog zu Bild 3.23*

Im Steuerfeld KZ werden die Variantennummern der entsprechenden Differenzenformeln für die partiellen Ableitungen $\partial P/\partial Z$ und $\partial^2 P/\partial Z^2$ abgelegt. Bild 3.35 zeigt die Steuerfelder KZ für die 2 Lagervarianten mit den Schmiertaschenanordnungen des Bildes Bild 3.23. Die Nummerierung der Varianten im Feld KZ ist identisch mit der Nummerierung der Varianten 1 bis 10 hier in der Dokumentation.

Feld der Punkttypen KZ

JX,JZ=1	2	3	4	5
1	9	1	1	7
2	9	1	1	7
3	9	1	1	7
4	9	1	1	7
5	0	2	1	7
6	0	0	2	7
7	10	0	0	2
8	9	3	0	0
9	9	1	3	0
10	9	1	1	7
11	10	0	0	0
12	10	0	0	0
13	10	0	0	0
14	10	0	0	0
15	9	1	1	7
16	9	1	1	7
17	9	1	1	7
18	9	1	1	7
19	9	1	1	7
20	9	1	1	7

Feld der Punkttypen KZ

JX,JZ=1	2	3	4	5	6	7	8	9	10
1	5	1	1	1	1	1	1	1	7
2	6	0	2	1	1	1	1	1	7
3	6	0	2	1	1	1	1	1	7
4	6	0	2	1	1	1	1	1	7
5	6	0	2	1	1	1	1	1	7
6	6	0	0	2	1	1	1	1	7
7	6	0	0	0	2	1	1	1	7
8	6	0	4	0	0	2	1	1	7
9	6	0	2	3	0	0	2	1	7
10	6	0	2	1	3	0	0	2	7
11	6	0	2	1	1	3	0	0	2
12	6	0	2	1	1	1	3	0	0
13	6	0	2	1	1	1	1	3	0
14	6	0	0	0	0	0	0	0	0
15	6	0	2	1	1	1	1	1	7
16	6	0	2	1	1	1	1	1	7
17	6	0	2	1	1	1	1	1	7
18	6	0	2	1	1	1	1	1	7
19	6	0	2	1	1	1	1	1	7
20	5	1	1	1	1	1	1	1	7

Bild 3.35: *Darstellung der Steuerfelder KZ für die 2 Varianten von Schmiertaschenanordnungen entsprechend Bild 3.23*

Zusätzlich zu den beiden Steuerfeldern KX und KZ erzeugt das Programm ein 3.Steuerfeld NG der Größe $N_Z \cdot N_X$, in dem es den einzelnen Flächenelementen der Schmierspaltfläche die Gleichungsnummern zuordnet, mit denen sie innerhalb des zu erzeugenden Gleichungssystems aufgeführt werden. Bild 3.36 zeigt die Steuerfelder NG für die 2 Lagervarianten mit den Schmiertaschenanordnungen des Bildes Bild 3.23.

Feld der Gleichungsnummern NG

JX,JZ=1	2	3	4	5
1	1	2	3	4	5
2	6	7	8	9	10
3	11	12	13	14	15
4	16	17	18	19	20
5	81	21	22	23	24
6	81	81	25	26	27
7	28	81	81	29	30
8	31	32	81	81	33
9	34	35	36	81	37
10	38	39	40	41	42
11	43	82	82	82	44
12	45	82	82	82	46
13	47	82	82	82	48
14	49	82	82	82	50
15	51	52	53	54	55
16	56	57	58	59	60
17	61	62	63	64	65
18	66	67	68	69	70
19	71	72	73	74	75
20	76	77	78	79	80

Feld der Gleichungsnummern NG

JX,JZ=1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	11	162	12	13	14	15	16	17	18	19
3	20	162	21	22	23	24	25	26	27	28
4	29	162	30	31	32	33	34	35	36	37
5	38	162	39	40	41	42	43	44	45	46
6	47	162	162	48	49	50	51	52	53	54
7	55	162	162	162	56	57	58	59	60	61
8	62	162	63	162	162	64	65	66	67	68
9	69	162	70	71	162	162	72	73	74	75
10	76	162	77	78	79	162	162	80	81	82
11	83	162	84	85	86	87	162	162	88	89
12	90	162	91	92	93	94	95	162	162	96
13	97	162	98	99	100	101	102	103	162	104
14	105	162	162	162	162	162	162	162	162	106
15	107	162	108	109	110	111	112	113	114	115
16	116	162	117	118	119	120	121	122	123	124
17	125	162	126	127	128	129	130	131	132	133
18	134	162	135	136	137	138	139	140	141	142
19	143	162	144	145	146	147	148	149	150	151
20	152	153	154	155	156	157	158	159	160	161

Bild 3.36: *Darstellung der Steuerfelder NG für die 2 Varianten von Schmiertaschenanordnungen entsprechend Bild 3.23*

Der Programmanwender bekommt das Steuerfeld KX nur in der Version des Bildes 3.23 zu sehen, ohne die internen Variantennummern für die Differenzenformeln, die die Darstellung an der Programmoberfläche nur unübersichtlicher machen würden. Die Steuerfelder KZ und NG bekommt er gar nicht zu Gesicht. Diese sind nur für den Programmierer bei einer Bearbeitung oder Weiterentwicklung des Programms von Interesse. Man kann sich diese Felder aber anzeigen lassen, wenn man in der Routine "PreProzessor" die Zeile

```
"c     call Kondru(NX,NZ,KX,KZ,NG,NGL)"
```

aktiviert, indem man den Buchstaben "c" am Zeilenanfang des Quelltexts entfernt und anschließend den Quelltext neu kompiliert. Nach dem Neustart des Programms werden dann die Steuerfelder KX, KZ und NG vollständig angezeigt, wenn im PreProzessor das Hauptmenü "Anordnung der Schmiertaschen festlegen" verlassen wird.

Die Vervollständigung des Steuerfeldes KX mit den Variantennummern und die Erzeugung der Steuerfelder KZ und NG erledigt die Routine "Muster", so dass diese Daten für eine zügige Berechnung der Koeffizientenmatrix des Gleichungssystems während der Hauptrechnung zur Verfügung stehen.

3.4.1.7 Erzeugung der Koeffizientenmatrix K(N_{Glei},N_{Glei}) und der rechten Seiten R(N_{Glei}) für die Berechnung der Druckverteilung im Schmierspalt

Durch die 8 Varianten für die Ableitungen in X-Richtung und die 10 Varianten für die Ableitungen in Z-Richtung gibt es 80 verschiedene Varianten von Differenzengleichungen zur Berechnung der Druckverteilung im Schmierspalt. Mit Hilfe der Steuerfelder KX, KZ und NG kann für jedes Feldelement des Schmierspalts die richtige lineare Differenzengleichung aufgestellt werden.

Nachfolgend soll als Beispiel eine Variante der aufzustellenden Differenzengleichungen gezeigt werden:

Ausgangsgleichung ist die lineare bzw. linearisierte Differentialgleichung der Form

$$\frac{\partial^2 P}{\partial X^2} + A_1 \cdot \frac{\partial^2 P}{\partial Z^2} + A_2 \cdot \frac{\partial P}{\partial X} + A_3 \cdot \frac{\partial P}{\partial Z} + A_4 \cdot P = R \tag{3.006}$$

Für den häufigsten Fall, nämlich für die Variante mit KX(J_Z,J_X)=1 und KZ(J_Z,J_X)=1, lautet die Differenzengleichung, die die Differentialgleichung approximiert,

$$\begin{aligned}
&\left(\frac{A_1(J_Z,J_X)\cdot 6}{6\cdot\Delta Z^2} - \frac{A_4(J_Z,J_X)\cdot 3}{6\cdot\Delta Z}\right) \cdot P(J_Z-1,J_X) + \\
&+\left(\frac{6}{6\cdot\Delta X^2} - \frac{A_3(J_Z,J_X)\cdot 3}{6\cdot\Delta X}\right) \cdot P(J_Z,J_X-1) + \\
&+\left(-\frac{12}{6\cdot\Delta X^2} - \frac{A_1(J_Z,J_X)\cdot 12}{6\cdot\Delta Z^2}\right) \cdot P(J_Z,J_X) + \\
&+\left(\frac{6}{6\cdot\Delta X^2} + \frac{A_3(J_Z,J_X)\cdot 3}{6\cdot\Delta X}\right) \cdot P(J_Z,J_X+1) + \\
&+\left(\frac{A_1(J_Z,J_X)\cdot 6}{6\cdot\Delta Z^2} + \frac{A_4(J_Z,J_X)\cdot 3}{6\cdot\Delta Z}\right) \cdot P(J_Z+1,J_X) = R(J_Z,J_X)
\end{aligned} \tag{3.087}$$

Wenn die zu berechnende Lagervariante ausnahmsweise mal keine Schmiertaschen besitzt, dann wird das lineare Gleichungssystem zur Berechnung der Druckverteilung P(N_Z,N_X) ausschließlich aus $N_X \cdot N_Z = N_{Glei} = N_{Glei1}$ Gleichungen gebildet, die die lineare bzw. linearisierte Differentialgleichung approximieren. Die Routine "KoMa1" erzeugt dazu die ungepackte Matrix K(N_{Glei1},N_{Glei1}) und den Vektor der rechten Seiten R(N_{Glei1}), womit das Gleichungssystem vollständig beschrieben ist und durch ein nummerisches Lösungsverfahren z.B. einen Gauss-Algorithmus mit Pivotisierung durch Routine "SIMQ" gelöst werden kann. Die Routine "KoMa1_pack" erzeugt analog dazu die Felder Kv, Kc und Kp und den Parameter M, die gemeinsam die entsprechende gepackte Matrix zur ungepackten Matrix K darstellen. Standardmäßig werden die gepackten Matrizen verwendet und mit dem GMRES-Verfahren mit ILU-Konditionierung gelöst, weil dieses wesentlich schneller arbeitet. Die Verwendung der ungepackten Matrizen und ihre Lösung mit einem Gauss-Algorithmus werden nur noch zur Kontrolle verwendet im Rahmen der Entwicklung und Fehlersuche. Ausführlicher dazu siehe Abschnitt 3.4.5.

3.4.2 Die Volumenbilanz einer Schmiertasche

Üblicherweise sind Gleitlager mit Schmiertaschen bzw. Schmiernuten versehen, über die Schmiermittel zu- bzw. auch abgeleitet werden kann. Diese sind dann über Verbindungsleitungen mit einer oder mehreren Schmiermittelpumpen verbunden. Schmiertaschen belegen einen Teil der Spaltfläche des Lagers zwischen Welle und Lagerschale. Da hier die Spalthöhe wesentlich größer ist als im eigentlichen Schmierspalt, kann hier die Newtonsche Reibung in der Flüssigkeit vernachlässigt werden und es gilt nicht die Reynoldssche Gleichung. Dieser Raum kann als ein reibungsverlustfreier hydrostatischer Raum aufgefasst werden, für den "lediglich" eine Volumenbilanz der durchströmenden Schmierflüssigkeit aufgestellt werden muss.

HINWEIS: *Die nachfolgenden Darstellungen beschreiben das erweiterte Modell eines Schmiermittel-Gas-Gemischs im Schmierspalt. Beim klassischen Fall vereinfachen sich die Gleichungen, indem der Füllungsgrad konstant auf F=1 gesetzt und damit das Flüssigkeitsvolumen identisch ist mit dem Spaltvolumen wird. So können für den klassischen Fall die entsprechenden Gleichungen leicht abgeleitet werden.*

Auch in den Schmiertaschen ist der Schmiermitteldruck zunächst unbekannt und muss im Rahmen der Hauptrechnung mit berechnet werden. Dazu ist die Bilanz des Flüssigkeitsvolumens jeder Schmiertasche aufzustellen. Die Gleichung (2.676) gibt die dimensionslose Bilanz des Flüssigkeitsvolumens einer Schmiertasche an.

$$Q_{Ta}(J_{Ta}) - Q_{TaRand}(J_{Ta}) = \frac{\partial Vol_{FlTa}(J_{Ta})}{\partial T} \tag{2.676}$$

Dabei sind Q_{Ta} der Schmiermittelstrom aus den angeschlossenen Verbindungleitungen in die Schmiertasche J_{Ta}, Q_{TaRand} der Schmiermittelstrom über den Schmiertaschenrand in den Schmierspalt, gemäß Gleichung (2.667), und $\partial Vol_{FlTa}/\partial T$ die Änderung des Flüssigkeitsvolumens in der Schmiertasche, gemäß Gleichung (2.674). Die Parameter Q_{TaRand} und $\partial Vol_{FlTa}/\partial T$ resultieren aus einer Integration über den Taschenrand bzw. über die Taschenfläche. Um diese Gleichungen in das zu berechnende lineare Gleichungssystem intergieren zu können, sind diese Integrale durch geeignete lineare Summenformeln zu approximieren. Dazu kann die bereits eingeführte Diskretisierung der Spaltfläche in $N_X \cdot N_Z$ Flächenelemente genutzt werden.

Zunächst soll das an einer Schmiertasche erläutert werden, die aus nur einem Flächenelement (J_Z,J_X) besteht.

3.4.2.1 Die Schmiermittelströme über die Ränder eines Flächenelements

Betrachten wir zunächst eine Schmiertasche, die nur aus dem Flächenelement $\Delta X \cdot \Delta Z$ besteht an der Stelle (J_Z,J_X). In Bild 3.37 sind die diskretisierten Schmiermittelströme und der Druckverlauf P in der Umgebung des Elements skizziert.

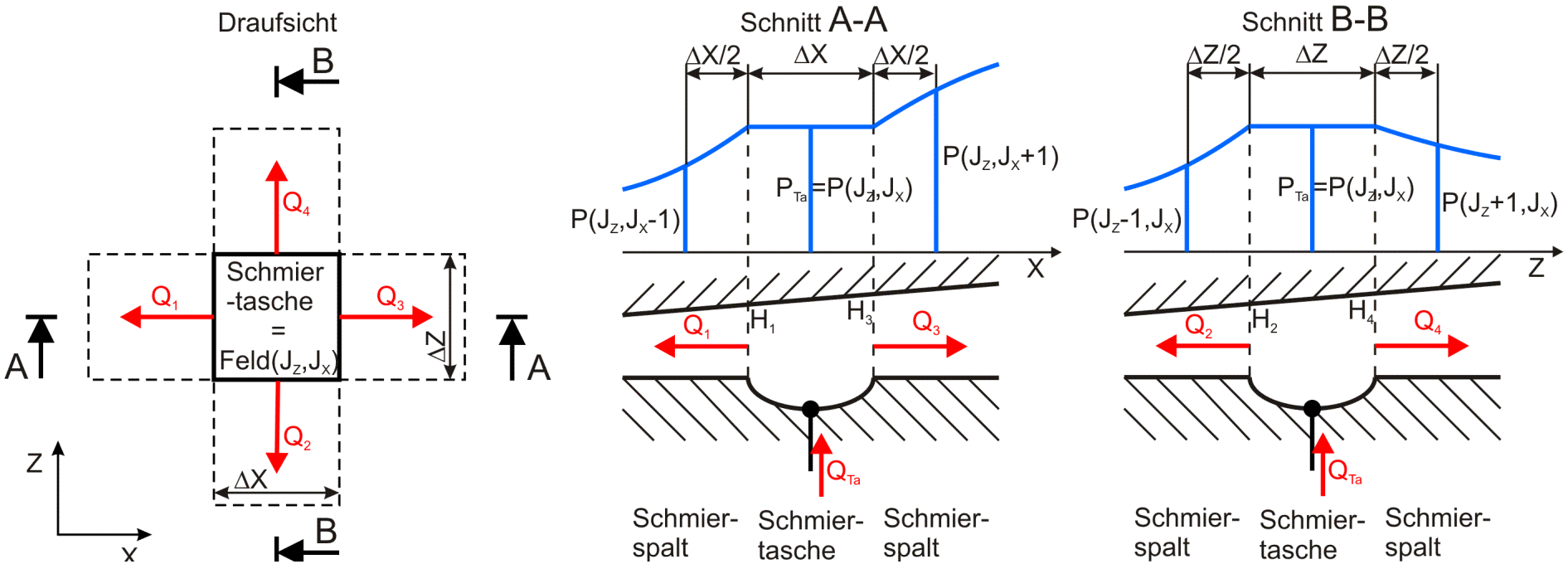


Bild 3.37: *Skizzen zu den Schmiermittelströmen über den Rand einer Schmiertasche, die nur aus einem Flächenelement besteht*

Der Schmiermittelstrom Q_{TaRand} über den Rand der Schmiertasche ist gemäß Gleichung (2.667) das Ringintegral über den Schmiertaschenrand.

$$(2.667)\qquad Q_{TaRand} = -\frac{1}{B^2}\oint_{Rand}\frac{H^3}{6}\cdot\frac{\partial P}{\partial Z}\cdot dX + \oint_{Rand}\frac{H^3}{6}\cdot\frac{\partial P}{\partial X}\cdot dZ - \frac{2\cdot\Omega}{\pi}\cdot\frac{P_{Ta}}{P_{Ta}+C}\cdot\oint_{Rand} H\cdot dZ$$

Er lässt sich für ein Flächenelement $\Delta X\cdot\Delta Z$ approximieren durch die Summe der 4 Teilströme.

$$(3.090)\qquad Q_{TaRand} \approx Q_1 + Q_2 + Q_3 + Q_4$$

Für die Teilströme lassen sich jetzt analog Gleichung (2.667) die Näherungsgleichungen angeben.

$$(3.091)\qquad Q_1 = -\frac{H_1^3\cdot\Delta Z}{6}\cdot\frac{\partial P}{\partial X} + \frac{2\cdot\Omega\cdot H_1\cdot B\cdot\Delta Z}{\pi}\cdot\frac{P_{Ta}}{P_{Ta}+C}$$

$$(3.092)\qquad Q_2 = -\frac{H_2^3\cdot\Delta X}{B^2\cdot 6}\cdot\frac{\partial P}{\partial Z}$$

$$(3.093)\qquad Q_3 = \frac{H_3^3\cdot\Delta Z}{6}\cdot\frac{\partial P}{\partial X} - \frac{2\cdot\Omega\cdot H_3\cdot\Delta Z}{\pi}\cdot\frac{P_{Ta}}{P_{Ta}+C}$$

$$(3.094)\qquad Q_4 = -\frac{H_4^3\cdot\Delta X}{B^2\cdot 6}\cdot\frac{\partial P}{\partial Z}$$

Die partiellen Ableitungen des Drucks in den 4 Formeln sind jeweils die Druckanstiege direkt am jeweiligen Schmiermittelrand. Sie können näherungsweise angegeben werden durch die 4 Formeln

$$(3.095)\qquad \frac{\partial P}{\partial X} = \frac{2\cdot\left[P_{Ta} - P(J_Z, J_X - 1)\right]}{\Delta X} \quad \text{bzw.} \quad \frac{\partial P}{\partial X} = \frac{2\cdot\left[P(J_Z, J_X + 1) - P_{Ta}\right]}{\Delta X}$$

und

$$(3.096)\qquad \frac{\partial P}{\partial Z} = \frac{2\cdot\left[P_{Ta} - P(J_Z - 1, J_X\right]}{\Delta Z} \quad \text{bzw.} \quad \frac{\partial P}{\partial Z} = \frac{2\cdot\left[P(J_Z + 1, J_X) - P_{Ta}\right]}{\Delta Z}$$

In den Gleichungen (3.091) und (3.093) befindet sich noch ein nichtlinearer Ausdruck $P_{Ta}/(P_{Ta}+C)$. Analog zu den Linearisierungen der erweiterten Reynoldsschen Differentialgleichung wird dieser ersetzt durch folgenden linearen Ausdruck:

$$(3.097)\qquad \frac{P_{Ta}}{P_{Ta}+C} \approx \frac{C}{(Pn_{Ta}+C)^2}\cdot P_{Ta} + \frac{Pn_{Ta}^2}{(Pn_{Ta}+C)^2}$$

Nun können alle 4 Teilströme durch lineare Näherungsformeln geschrieben werden.

$$(3.098)\qquad Q_1 = \left(\frac{H_1^3\cdot\Delta Z}{3\cdot\Delta X} + \frac{2\cdot\Omega\cdot H_1\cdot\Delta Z\cdot C}{\pi\cdot(Pn_{Ta}+C)^2}\right)\cdot P_{Ta} - \frac{H_1^3\cdot\Delta Z}{3\cdot\Delta X}\cdot P(J_Z, J_X - 1) + \frac{2\cdot\Omega\cdot H_1\cdot\Delta Z\cdot Pn_{Ta}^2}{\pi\cdot(Pn_{Ta}+C)^2}$$

$$(3.099)\qquad Q_2 = \frac{H_2^3\cdot\Delta X}{B^2\cdot 3\cdot\Delta Z}\cdot P_{Ta} - \frac{H_2^3\cdot\Delta X}{B^2\cdot 3\cdot\Delta Z}\cdot P(J_Z - 1, J_X)$$

$$(3.100)\qquad Q_3 = \left(\frac{H_3^3\cdot\Delta Z}{3\cdot\Delta X} + \frac{2\cdot\Omega\cdot H_3\cdot\Delta Z\cdot C}{\pi\cdot(Pn_{Ta}+C)^2}\right)\cdot P_{Ta} - \frac{H_3^3\cdot\Delta Z}{3\cdot\Delta X}\cdot P(J_Z, J_X + 1) + \frac{2\cdot\Omega\cdot H_3\cdot\Delta Z\cdot Pn_{Ta}^2}{\pi\cdot(Pn_{Ta}+C)^2}$$

$$(3.101)\qquad Q_4 = \frac{H_4^3\cdot\Delta X}{B^2\cdot 3\cdot\Delta Z}\cdot P_{Ta} - \frac{H_4^3\cdot\Delta X}{B^2\cdot 3\cdot\Delta Z}\cdot P(J_Z + 1, J_X)$$

Mit den 4 Gleichungen (3.098) bis (3.101) kann man jetzt den Schmiermittelstrom Q_{TaRand} über den Schmiertaschenrand als lineare Gleichung schreiben mit den 6 Unbekannten $P(J_Z\text{-}1,J_X)$, $P(J_Z,J_X\text{-}1)$, $P(J_Z,J_X\text{+}1)$, $P(J_Z\text{+}1,J_X)$, P_{Ta} und Q_{TaRand} in folgender Form.

$$(3.102) \qquad Q_{TaRand} \approx K_1 \cdot P(J_Z-1,J_X)+K_2 \cdot P(J_Z,J_X-1)+K_3 \cdot P(J_Z,J_X+1)+K_4 \cdot P(J_Z,J_X+1)+K_5 \cdot P_{Ta}+K_6$$

Die Koeffizienten der Gleichung werden hier nicht ausgeschrieben, da diese Formel zu umfangreich und unübersichtlich wird. Sie wurde in dieser Form auch nie ausgeschrieben. Im Programm werden diese Koeffizienten rekursiv über mehrere Hilfsvariable wesentlich rationeller bestimmt.

3.4.2.2 Änderung des Schmiermittelvolumens in einem Flächenelement

Durch eine Änderung der Spalthöhe $\partial H(J_Z,J_X)/\partial T$ und/oder eine Druckänderung $\partial P_{Ta}/\partial T$ kann sich das Flüssigkeitsvolumens Vol_{FlTa} in der Schmiertasche ändern, was in die Volumenbilanz des Schmiertaschenelements eingehen muss.

In Anlehnung an die Gleichung (2.674) ist die dimensionslose Änderung des Flüssigkeitsvolumens in einem Schmiertaschenelement über die Zeit T näherungsweise gegeben durch

$$(3.103) \qquad \frac{\partial \Delta Vol_{FlTa}}{\partial T} \approx \frac{1}{\pi} \cdot \frac{\partial F_{Ta}}{\partial T} \cdot H(JZ,JX) \cdot \Delta X \cdot \Delta Z + F_{Ta} \cdot \frac{\partial H}{\partial T} \cdot \Delta X \cdot \Delta Z$$

In der Hauptrechnung soll zunächst der Druck in der Schmiertasche berechnet werden. Deshalb ist die Gleichung (3.103) auf die Angabe von Druck umzustellen. Der örtliche Füllungsgrad F_{Ta} des Schmierspalts ist gegeben durch den Druck P_{Ta} in der Schmiertasche gemäß Gleichung (2.614).

$$(3.104) \qquad F_{Ta} = \frac{P_{Ta}}{P_{Ta}+C}$$

Die Ableitung nach der Zeit T ist dann gegeben durch

$$(3.105) \qquad \frac{\partial F_{Ta}}{\partial T} = \frac{C}{(P_{Ta}+C)^2} \cdot \frac{\partial P_{Ta}}{\partial T}$$

Würde man die Gleichungen (3.104) und (3.105) in die Gleichung (3.102) einsetzen, würde diese wieder nichtlinear, was nicht im Sinne des Lösungsverfahrens ist, nämlich ein lineares Gleichungssystem zu erzeugen. Deshalb werden auch diese Gleichungen zunächst wieder in gewohnter Weise durch linearisierte Näherungen ersetzt.

$$(3.106) \qquad F_{Ta} \approx \frac{C}{(Pn_{Ta}+C)^2} \cdot P_{Ta} + \frac{Pn_{Ta}^2}{(Pn_{Ta}+C)^2}$$

und

$$(3.107) \qquad \frac{\partial F_{Ta}}{\partial T} \approx \left[-\frac{2 \cdot C}{(Pn_{Ta}+C)^3} \cdot \frac{\partial Pn_{Ta}}{\partial T} + \frac{C \cdot (2 \cdot Pn_{Ta} - P_{Ta}(J_T-1)}{(Pn_{Ta}+C)^2 \cdot P_{Ta}(J_T-1) \cdot \Delta T} \right] \cdot P_{Ta} + \\ + \frac{2 \cdot C \cdot Pn_{Ta}}{(Pn_{Ta}+C)^3} \cdot \frac{\partial Pn_{Ta}}{\partial T} - \frac{C \cdot Pn_{Ta}^2}{(Pn_{Ta}+C)^2 \cdot P_{Ta}(J_T-1) \cdot \Delta T}$$

Dabei wird die Ableitung von Pn_{Ta} über die Zeit näherungsweise berechnet durch die Gleichung

$$(3.108) \qquad \frac{\partial Pn_{Ta}}{\partial T} \approx \frac{Pn_{Ta}}{P(J_T-1)} \cdot \frac{Pn_{Ta}-P(J_T-1)}{\Delta T}$$

Pn_{Ta} ist hier die vorhergehende Näherung für den Druck P_{Ta} für den aktuellen Zeitpunkt J_T. $P_{Ta}(J_T\text{-}1)$ ist der Schmiertaschendruck zum vorhergehenden Zeitpunkt. Ausführlich erläutert ist die Linearisierung in [20, Abschnitt 7.3].

Damit lässt sich auch die Änderung des Schmierflüssigkeitsvolumens in der Schmiertasche durch einen linearen Ausdruck angeben in der Form

$$(3.109) \qquad \frac{\partial Vol_{FlTa}}{\partial T} \approx K_7 \cdot P_{Ta} + K_8$$

Die Koeffizienten der Gleichung werden auch hier nicht ausgeschrieben, da diese Formel zu umfangreich und unübersichtlich wird. Sie wurde in dieser Form auch nie ausgeschrieben. Im Programm werden diese Koeffizienten rekursiv über mehrere Hilfsvariable wesentlich rationeller bestimmt.

3.4.2.3 Volumenbilanz einer Schmiertasche, die nur aus einem Flächenelement besteht

Mit den Formeln der beiden vorhergehenden Abschnitte lässt sich jetzt die Volumenbilanz der Schmiermittelströme in einem Flächenelement einer Schmiertasche angeben als lineare Gleichung in der Form

$$(3.110) \qquad K_1 \cdot P(J_Z-1,J_X)+K_2 \cdot P(J_Z,J_X-1)+K_3 \cdot P(J_Z,J_X+1)+K_4 \cdot P(J_Z,J_X+1)+(K_5+K_7) \cdot P_{Ta}-Q_{Ta} \approx -K_6-K_8$$

3.4.2.4 Volumenbilanz einer Schmiertasche, die mehrere der Flächenelemente der diskretisierten Spaltfläche überdeckt

Bild 3.38 zeigt anhand der Steuermatrix KX die Ausdehnung der ersten Schmiertasche J_{Ta}=1 über mehrere Flächenelemente $\Delta X \cdot \Delta Z$.

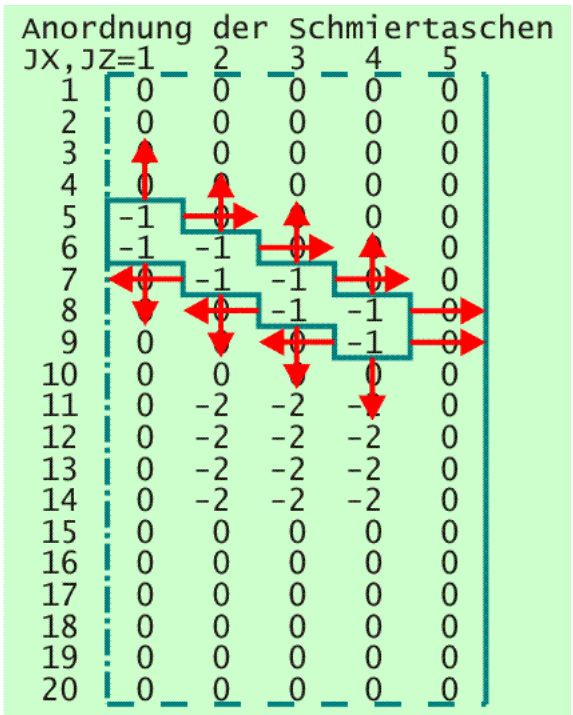


Bild 3.38: *Diskretisierte Schmiermittelströme über den Rand einer Schmiertasche*

Die roten Pfeile sollen die diskretisierten Teilölströme aus der Schmiertasche in den eigentlichen Schmierspalt darstellen. Rechnerisch wäre es korrekt, wenn man sich eine großflächige Schmiertasche aus mehreren elementaren Schmiertaschen, bestehend aus je einem Flächenelement, vorstellt und die Volumenbilanzen der Teilflächen gemäß Formel (3.109) aufsummiert. Die internen Schmiermittelströme über die Elementgrenzen würden sich dann gegenseitig aufheben.

Programmintern werden aber nur die erforderlichen Ströme am Taschenrand berechnet. Hier bedient sich das Programm wieder der Steuermatrix KX, indem es abfragt, ob neben dem aktuell behandelten Flächenelement auch noch Flächenelemente von Schmiertaschen oder der Lagerrand liegen.

So ist für jede der N_{Ta} Schmiertaschen genau eine weitere lineare Gleichung in das Gleichungssystem der Hauptrechnung zu schreiben.

Die Routine "KoMa3" bzw. die Routine "KoMa3_pack" erledigt das. Dabei kommen die Drücke $P_{Ta}(J_{Ta})$ mit J_{Ta}=1 bis N_{Ta} als weitere Unbekannte in das Gleichungssystem. Der Ölstrom Q_{Ta} erscheint nicht explizit als eine Unbekannte im Gleichungssystem. An ihrer Stelle werden der bzw. die entsprechenden Ölströme $Q_{Ve}(J_{Ve})$ aus den Verbindungsleitungen eingetragen, mit denen die jeweilige Schmiertasche verbunden ist.

3.4.3 Vervollständigen des linearen Gleichungssystems durch weitere Gleichungen, die das periphere Schmiermittel-Versorgungssystem beschreiben

Das periphere Schmiermittel-Versorgungssystem kann mehrere Schmiermittelpumpen besitzen, die durch mehrere Verbindungsleitungen auf sehr unterschiedliche Weise mit den einzelnen Schmiertaschen verbunden sind. In diesen Leitungen ist jeweils ein Gerät angeordnet, dass den Schmiermittelstrom durch die Leitung beeinflusst. Im Rahmen der Hauptrechnung zum Lager müssen simultan zur Berechnung der Druckverteilung im Schmierspalt diverse Drücke und Schmiermittelströme als primäre Ergebnisparameter berechnet werden. Die notwendigen Ergebnisparameter für das Schmiersystem, aus denen dann auch alle anderen Daten ermittelt werden können, sind neben den Schmiertaschendrücken P_{Ta} die aktuellen Schmiermittelströme Q_{Ve} durch die einzelnen Verbindungsleitungen, die Pumpenströme Q_{Pu} und die Pumpendrücke P_{Pu}. Dazu sind die Gerätekennlinien und die Verknüpfungen der Elemente des Schmiersystems durch geeignete lineare Gleichungen abzubilden, die dem linearen Gleichungssystem zugefügt werden.

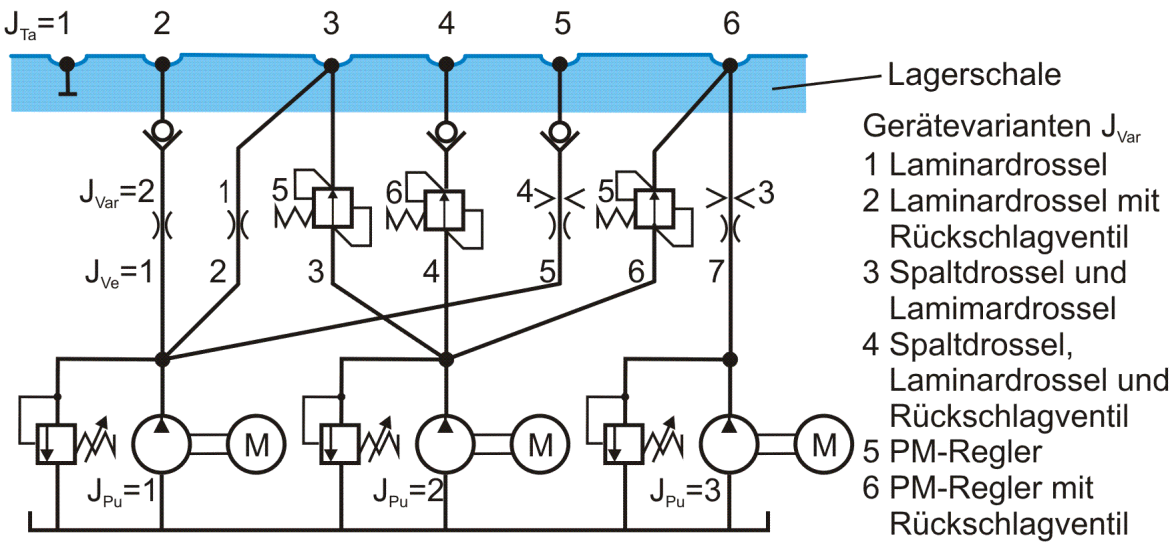


Bild 4.025: *Prinzipskizze einer möglichen Variante des peripheren Universal-Schmiersystems*

Bild 4.025 zeigt beispielhaft die mögliche Komplexität eines solchen Schmiermittel-Versorgungssystems. Bis auf die nichtlineare Kennlinie der Spaltdrossel sind die Gerätekennlinien recht einfache lineare Funktionen. Allerdings bestehen diese teilweise aus mehreren Ästen, aus denen sie zusammengesetzt sind. Die eigentliche Herausforderung der Programmierung des Schmiermittel-Versorgungssystems ist die Abbildung der Verknüpfungen der verschiedenen Elemente. So muss das Programm befähigt werden, anhand der verfügbaren Steuerparameter für jeden konkreten Fall automatisch die richtigen Gleichungen auszuwählen und dementsprechend die Koeffizienten in die Matrix des Gleichungssystems einzutragen. Erschwert wird diese Aufgabe weiter dadurch, dass anschließend ein schnelles Lösungsverfahren verwendet werden soll, dass mit einer gepackten Matrix arbeitet (siehe Abschnitt 3.4.5.1). Die gepackte Matrix macht das Ganze noch unübersichtlicher. Der Aufwand lohnt sich aber, da mit diesem Verfahren gegenüber dem Verfahren mit ungepackter Matrix eine Verkürzung der Rechenzeit um den Faktor 1000 erreicht wird.

Die Lösung dieser Teilaufgabe ist in der Routine KoMa4 bzw. KoMa4_pack realisiert. Die Erzeugung dieser Routinen war für den Autor eine der schwierigsten Teilaufgaben, weil es hier besonders schwierig ist, den Überblick über die vielen verschiedenen logischen Verknüpfungen zu behalten. Beruhigend an der Programmierung dieses Teilsystems ist, dass die Überprüfung der Ergebnisse auf ihre Richtigkeit recht einfach ist, weil man mit einfachen Gleichungen das Ergebnis manuell nachrechnen kann. Wenn allerdings ein Fehler festgestellt wird, ist die Fehlersuche recht aufwendig.

Das mag den Anwender nicht weiter interessieren, wenn einmal das Programm erzeugt und ausführlich getestet wurde. Da der Autor das Programm aber nicht nur als Black Box zur allgemeinen Nutzung übergeben möchte, sondern es auch zulässt und dazu anregen möchte, das Programm zu ergänzen und weiterzuentwickeln, soll hier die Programmierung in möglichst verständlicher Form offengelegt werden.

Zur Lösung des Problems ist der Autor in 3 Schritten vorgegangen:

1. Es wurde der Programmablauf in einer programmähnlichen und noch recht gut lesbaren Schreibweise entworfen. Siehe Abschnitt 3.4.3.2.

2. Die Routine KoMa4 wurde zunächst für den einfacheren Fall, mit ungepackter Koeffizientenmatrix, geschrieben und getestet.

3. Mit der Routine KoMa4_pack wurde die Routine aus Koma4 umgeschrieben für den unübersichtlicheren Fall mit einer gepackten Matrix.

Nachdem das Problem erfolgreich mit gepackter Matrix gelöst war, wurde die Routine KoMa4 und ihre Einbindung in die übergeordneten Routinen nicht etwa entfernt, sondern sie wurde nur innerhalb des Programms deaktiviert. Im Falle eines auftretenden Fehlers oder zur Weiterentwicklung des Programms kann diese Routine leicht wieder aktiviert werden. Siehe dazu Abschnitt 3.4.5.4.

3.4.3.1 Die Linearisierung der Gleichung für die Kennlinie einer Spaltdrossel und einer Kapillare in Reihenschaltung

Die Betriebskennlinie einer Spaltdrossel (Blende) in Reihe mit einer Laminardrossel (Kapillare) ohne Rückschlagventil wird durch die 2 Gleichungen abgebildet (siehe dazu auch den Abschnitt 2.2.6.2.2)

$$(2.705) \qquad Q_{Ve} = -\frac{Ccp \cdot Cbl}{2} + \sqrt{\left(\frac{Ccp \cdot Cbl}{2}\right)^2 + Cbl \cdot P_{VeVer}} \qquad \text{für } P_{VeVer} \geq 0$$

$$(2.706) \qquad Q_{Ve} = +\frac{Ccp \cdot Cbl}{2} - \sqrt{\left(\frac{Ccp \cdot Cbl}{2}\right)^2 - Cbl \cdot P_{VeVer}} \qquad \text{für } P_{VeVer}<0$$

mit

Ccp - dimensionsloser Widerstandsbeiwert des laminaren Strömungswiderstandes

Cbl - dimensionsloser Blendenbeiwert der Spaltdrossel

P_{VeVer} - dimensionsloser Druckverlust über die Drosseln $P_{VeVer}=P_{Pu}-P_{Ta}$

Das ist bisher die einzige Kennlinie der implementierten, stromregelnden Geräte in den Verbindungsleitungen, die nicht linear ist. Um diese Kennlinie in das lineare Gleichungssystem der Hauptrechnung einbeziehen zu können, muss auch diese Gleichung analog zur Linearisierung der erweiterten Reynoldsschen Gleichung linearisiert werden und kann so ebenfalls iterativ berechnet werden. Das bietet sich an, da in der Berechnung ohnehin bereits mehrere Iterationszyklen existieren, in denen auch diese Iteration erfolgen kann, so dass das keinen nennenswerten zusätzlichen Rechenaufwand bedeutet.

Auch hier wird für die Linearisierung ein Ansatz mit den ersten 2 Gliedern der Taylorreihe verwendet

$$(3.113) \qquad Q_{Ve}(P_{VeVer}) \approx Q_{Ve}(Pn_{VeVer}) + \frac{\partial Q_{Ve}(Pn_{VeVer})}{\partial P_{VeVer}} \cdot (P_{VeVer} - Pn_{VeVer})$$

Angewendet auf die beiden Gleichungen (2.705) und (2.706) ergibt das die Gleichungen

$$(3.114) \qquad Q_{Ve}(P_{VeVer}) \approx -C_1 + C_2 - C_3 \cdot Pn_{VeVer} + C_3 \cdot P_{VeVer} \text{ für } P_{VeVer} \geq 0$$

und

$$(3.115) \qquad Q_{Ve}(P_{VeVer}) \approx +C_1 - C_2 - C_3 \cdot Pn_{VeVer} + C_3 \cdot P_{VeVer} \text{ für } P_{VeVer} < 0$$

mit

$$(3.116) \qquad C_1 = \frac{Ccp \cdot Cbl}{2}$$

$$(3.117) \qquad C_2 = \sqrt{C_1^2 + Cbl \cdot |Pn_{VeVer}|} = \sqrt{C_1^2 + Cbl \cdot |Pn_{Pu} - Pn_{Ta}|}$$

$$(3.118) \qquad C_3 = \frac{Cbl}{2 \cdot C_2}$$

Mit der Definition der Funktion sign(x,y), wie sie in der Programmiersprache FORTRAN definiert ist, lassen sich die beiden Gleichungen (3.114) und (3.115) zusammenfassen.

$$(3.119) \qquad Q_{Ve}(P_{VeVer}) \approx \text{sign}(-C_1 + C_2, Pn_{VeVer}) - C_3 \cdot Pn_{VeVer} + C_3 \cdot P_{VeVer}$$

HINWEIS: *Die Funktion sign(x,y) bedeutet in FORTRAN folgendes: Die Funktion liefert als Ergebnis den absoluten Betrag der Variablen x mit dem Vorzeichen der Variablen y. Lit.[8,Seite 11-20]*

Für den Fall, dass die Schmiermittelpumpe aktuell als Pumpe mit konstantem Pumpendruck arbeitet (druckgeregelte Pumpe), d.h. $Pn_{Pu}=P_{Pu}=P_{PuMax}$, gilt

$$(3.120) \qquad Pn_{VeVer} = P_{PuMax} - Pn_{Ta}$$

und

$$(3.121) \qquad P_{VeVer} = P_{PuMax} - P_{Ta}$$

Eingesetzt in die Gleichung (3.119) ergibt das die linearisierte Gleichung für die Kennlinie

$$(3.122) \qquad Q_{Ve}(P_{Pu}, P_{Ta}) \approx \text{sign}(-C_1 + C_2, P_{PuMax} - Pn_{Ta}) + C_3 \cdot Pn_{Ta} - C_3 \cdot P_{Ta}$$

Für den Fall, dass die Schmiermittelpumpe aktuell als Pumpe mit konstantem Förderstrom arbeitet (Konstantpumpe), d.h. $Pn_{Pu}<P_{PuMax}$, gilt

(3.123) $$Pn_{VeVer} = Pn_{Pu} - Pn_{Ta}$$

und

(3.124) $$P_{VeVer} = P_{Pu} - P_{Ta}$$

Eingesetzt in die Gleichung (3.119) ergibt das die linearisierte Gleichung für die Kennlinie

(3.125) $$Q_{Ve}(P_{Pu},P_{Ta}) \approx sign(-C_1 + C_2, Pn_{Pu} - Pn_{Ta}) - C_3 \cdot (Pn_{Pu} - Pn_{Ta}) + C_3 \cdot P_{Pu} - C_3 \cdot P_{Ta}$$

3.4.3.2 Darstellung der Routine KoMa4 in einer programmähnlichen Schreibweise

Nachfolgend soll der Ablauf der Erzeugung der restlichen Gleichungen zur Abbildung des Schmiermittel-Versorgungssystems in einer programmähnlichen Form dargestellt werden. Es sind hier die **eigentlichen Befehle** in farbiger Schrift dargestellt, während die Kommentierung in der üblichen schwarzen Schrift erfolgt. Bei der Darstellung wurde zur Wahrung der Übersichtlichkeit weitgehend auf die Indizes zu den variablen Feldbezeichnungen verzichtet. Es wurde aber bereits die Reihenfolge der späteren Abarbeitung der Befehle in der Routine KoMa4 eingehalten.

Darstellung des Programmablaufs in KoMa4 in vereinfachter, lesbarer Form:

$J_2 = N_{Glei} + N_{Ta}$

$J_3 = N_{Glei} + N_{Ta} + N_{Ve}$ $= J_2 + N_{Ta}$

$J_4 = N_{Glei} + N_{Ta} + N_{Ve} + N_{Pu}$ $= J_3 + N_{Pu}$

Ergänzen der N_{Ve} Gleichungen Nr. J_2+1 bis J_2+N_{Ve}, die den Zusammenhang zwischen dem Schmiermittelstrom Q_{Ve} und dem Druckgefälle $P_{VeVer} = P_{Pu} - P_{Ta}$ in den N_{Ve} Verbindungsleitungen beschreiben. Das sind die Betriebskennlinien der jeweiligen Geräte in den Verbindungsleitungen: $Q_{Ve}=f(P_{VeVer}=P_{Pu}-P_{Ta})$ bzw. $Q_{Ve}=f(P_{VeVer}=P_{PuMax}-P_{Ta})$

Für J_{Ve} = 1 bis N_{Ve} tue folgendes:

Rechter horizontaler Ast der Kennlinie für alle Geräte mit Rückschlagventil

Wenn **Pn_{Pu} < Pn_{Ta}** und ein **Rückschlagventil** vorhanden ist, dann gilt:

Q_{Ve} = 0

Gehe weiter zu Marke 20

endif.

Wenn das Gerät in der Leitung eine **Kapillare** ist, dann gilt:

Wenn Pu_{Var} = 2, dann gilt (Druckgeregelte Pumpe)

Q_{Ve} = (P_{PuMax} -P_{Ta}) / Ccp (Gleichung 2.695)

Gehe weiter zu Marke 20

sonst gilt (Konstantpumpe)

Q_{Ve} = (P_{Pu} - P_{Ta}) / Ccp (Gleichung 2.695)

Gehe weiter zu Marke 20

endif.

endif.

Wenn das Gerät in der Leitung ein **PM-Regler** ist, dann gilt:

$P_0 = Q_{P1}$ / Cpm

Wenn $Pn_{Ta} \leq Pn_{Pu} - P_0$, dann gilt (linker horizontaler Ast der Kennlinie)

Q_{Ve} = 0 (Gleichung 2.723, 1.Fall)

Gehe weiter zu Marke 20

endif.

Wenn (P_{Pu} - P_0) < P_{Ta} < (P_{Pu} - P_S) , dann: (aufsteigender Ast der Kennlinie)

Wenn Pu_{Var} = 2, dann gilt (Druckgeregelte Pumpe)

$Q_{Ve} = Q_{P1} - Cpm \cdot (P_{PuMax} - P_{Ta})$ (Gleichung 2.723, 2.Fall)

sonst gilt (Konstantpumpe)

$Q_{Ve} = Q_{P1} - Cpm \cdot (P_{Pu} - P_{Ta})$ (Gleichung 2.723, 2.Fall)

endif.

Gehe weiter zu Marke 20

endif.

Wenn $Pn_{Ta} \geq$ (P_{Pu} - P_S), dann gilt: (absteigender Ast der Kennlinie)

Wenn Pu_{Var} = 2, dann gilt (Druckgeregelte Pumpe)

$Q_{Ve} = (P_{PuMax} - P_{Ta}) / Rpm$ (Gleichung 2.723, 3.Fall)

sonst gilt (Konstantpumpe)

$Q_{Ve} = (P_{Pu} - P_{Ta}) / Rpm$ (Gleichung 2.723, 3.Fall)

endif.

Gehe weiter zu Marke 20

endif.

endif.

Wenn das Gerät in der Leitung eine **Blende und eine Kapillare** in Reihe ist, dann gilt:

$C_1 = Ccp \cdot Cbl / 2$

$C_2 = sqrt (C_1^2 + Cbl \cdot |Pn_{Pu} - Pn_{Ta}|)$

$C_3 = Cbl / 2 / C_2$

Wenn Pu_{Var} = 2 ,

dann gilt (Druckgeregelte Pumpe)

$Q_{Ve} = sign (-C_1+C_2 , Pn_{Pu}-Pn_{Ta}) + C_3 \cdot Pn_{Ta} - C_3 \cdot P_{Ta}$

(Gleichung 3.122)

sonst gilt (Konstantpumpe)

$Q_{Ve} = sign (-C_1+C_2 , Pn_{Pu}-Pn_{Ta}) - C_3 \cdot (Pn_{Pu} - Pn_{Ta}) + C_3 \cdot P_{Pu} - C_3 \cdot P_{Ta}$

(Gleichung 3.125)

endif.

endif.

Marke 20

Ende der Do-Schleife.

HINWEIS: *Die Funktion sign(x,y) bedeutet in FORTRAN folgendes: Die Funktion liefert als Ergebnis den absoluten Betrag der Variablen x mit dem Vorzeichen der Variablen y. Lit.[8,Seite 11-20]*

Ergänzen der N_{Pu} Gleichungen Nr. J_3+1 bis J_3+N_{Pu} für die aktuellen Schmiermittelströme $Q_{Pu}(J_{Pu})$ der Pumpen:
Für J_{Pu} = 1 bis N_{Pu} tue folgendes:

Wenn Pu_{Var} = 1 , dann (Konstantpumpe)

$Q_{Pu} = QP_{uMax}$

Wenn Pu_{Var} = 2 , dann (Druckgeregelte Pumpe)

$Q_{Pu} (J_{Pu}) = Summe (Q_{Ve} (J_{Ve}))$ aller Leitungen Nr. J_{Ve} , die mit der Pumpe Nr. J_{Pu} verbunden sind

(Gleichung 2.730)

Ende der Do-Schleife.

Ergänzen der N_{Pu} Gleichungen Nr. J_4+1 bis J_4+N_{Pu} zur Begrenzung des Pumpendrucks auf P_{Pumax} bzw. des Pumpenstroms auf Q_{PuMax}:
Für J_{Pu} = 1 bis N_{Pu} tue folgendes:

Wenn Pu_{Var} = 2, dann gilt (Druckgeregelte Pumpe)

$P_{Pu} = P_{PuMax}$

sonst gilt (Konstantpumpe)

$Q_{PuMax} (J_{Pu}) = Summe(Q_{Ve}(J_{Ve}))$ aller Leitungen Nr. J_{Ve} , die mit der Pumpe Nr. J_{Pu} verbunden sind

(Gleichung 2.730)

endif

Ende der Do-Schleife.

Damit ist das Gleichungssystem eigentlich vollständig. Es wurden $N_{Glei}+N_{Ta}+N_{Ve}+2 \cdot N_{Pu} = N_{Glei1}$ linear unabhängige Gleichungen aufgestellt mit ebenso vielen unbekannten Variablen, für die genau eine Lösung zu erwarten ist. Mit einem Gauss-Algorithmus mit Pivotisierung (z.B. mit der Routine SIMQ) ist dieses Gleichungssystem lösbar.

Das Gleichungssystem soll aber mit dem GMRES-Verfahren mit ILU-Konditionierung gelöst werden. Dieses Verfahren fordert, dass die Hauptdiagonale der Matrix vollbesetzt ist, d.h. dass auf dieser Diagonale kein Koeffizient mit dem Wert Null auftritt. Diese Forderung ist aber bei diesem Gleichungssystem nicht immer erfüllt. Wenn für eine oder mehrere Pumpen Nr. J_{Pu} gilt: $Pu_{Var}(J_{Pu}) = 1$ (Konstantpumpe), dann wird im letzten Gleichungsblock die Gleichung (2.730) aufgestellt. In diesem Fall steht in der Hauptdiagonale der Gleichung Nr. J_4+J_{Pu} eine Null, weil die unbekannte Variable $P_{Ta}(J_{Pu})$ in dieser Gleichung nicht vorkommt. Das betrifft im oben dargestellten Programmablauf den Block der letzten N_{Pu} Gleichungen und hier den 2.Fall. Dem kann abgeholfen werden, indem eine der bereits vorhandenen anderen Gleichungen, die in der Spalte (J_4+J_{Pu}) einen von Null verschiedenen Koeffizienten aufweist, zu der Gleichung addiert wird. Geeignete Gleichungen findet man im Block der Gleichungen Nr. J_2+1 bis J_2+N_{Ve}. Das sind die Gleichungen der Gerätekennlinien. Deshalb ist nun noch für diesen speziellen Fall die Routine KoMa4 entsprechend zu ergänzen:

Für J_{Ve} = 1 bis N_{Ve} tue folgendes:

Wenn PuVar =1 und ($Pn_{Pu} \geq Pn_{Ta}$ oder das Gerät in der Leitung kein Rückschlagventil aufweist),

dann addiere die Gleichung Nr. J_2+N_{Ve}:

$Q_{Ve}(J_{VE})=f(P_{Pu}(Ve(1,J_{Ve}))-P_{Ta}(Ve(2,J_{Ve})))$ (Funktion der Gerätekennlinie)

zur Gleichung Nr. J_4+N_{Pu}:

$Q_{PuMax} (J_{Pu}) = Summe(Q_{Ve}(J_{Ve}))$ aller Leitungen Nr. J_{Ve} , die mit der Pumpe Nr. J_{Pu} verbunden sind

endif.

Damit ist der Programmablauf für die Berechnung der Koeffizientenmatrix des Gleichungssystems und der rechten Seiten komplett.

Mit diesem Entwurf des Programmablaufs kann der Quellcode für die Routine KoMa4 für eine ungepackte Koeffizientenmatrix geschrieben und getestet werden. Den kommentierten Quellcode der Routine "KoMa4" findest Du in der Textdatei gleichen Namens "KoMa4.f" Anschließend kann daraus der Quellcode für die Routine "KoMa4_pack" abgeleitet werden und ebenfalls getestet werden. Dieses stufenweise Vorgehen erleichtert es, Programmfehler zu finden.

Jeder, der dieses Programm evtl. um einen neuen Gerätetyp in den Verbindungsleitungen erweitern möchte, muss die Routine KoMa4_pack bearbeiten. Die Routine ist Bestandteil des Kerns der Berechnungen und Programmierfehler können hier die gesamte Berechnung verderben. Deshalb ist hier mit besonderer Umsicht vorzugehen und die Ergebnisse sind anschließend eingehend zu überprüfen.

3.4.4 Übersicht der Struktur der Matrix und der rechten Seiten des linearen Gleichungssystems

Die **Koeffizientenmatrix K** bekommt der Anwender normaler Weise nicht zu Gesicht. Um den prinzipiellen Aufbau dieser Matrix einmal zu zeigen, soll aber hier diese Matrix an einem ganz kleinen Beispiel dargestellt werden:

Angenommen wird ein Gleitschuh, dessen Spaltfläche in nur 5·5 Flächenelemente aufgeteilt ist, gemäß Bild 3.40.

```
Anordnung der Schmiertaschen (Steuerfeld KX)
JX,JZ=1   2   3   4   5
  1   0   0   0   0   0
  2   0   0   0   0   0
  3   0  -1   0  -2   0
  4   0   0   0   0   0
  5   0   0   0   0   0
```

***Bild 3.40**: Diskretisierte Spaltfläche eines Gleitschuhs, dargestellt anhand des Steuerfeldes KX*

In dem Schmierspalt sind 2 Schmiertaschen angeordnet, die jeweils nur aus einem Flächenelement bestehen. Jede Schmiertasche ist mit je einer Verbindungsleitung mit je einer Schmiermittelpumpe verbunden. In den Verbindungsleitungen ist je eine Kapillare ohne Rückschlagventil angeordnet.

Daraus ergeben sich folgende Daten für das Gleichungssystem:

N_{Ta}	= 2	Anzahl der Schmiertaschen
N_{Ve}	= 2	Anzahl der Verbindungsleitungen
N_{Pu}	= 2	Anzahl der Schmiermittelpumpen
N_{Glei}	= $N_X \cdot N_Z - N_{Ta}$ = 5 · 5 -2= = 23	Anzahl der Feldelemente (Gitterpunkte), für die der Schmierfilmdruck berechnet werden muss
N_{Glei1}	= $N_{Glei}+N_{Ta}+N_{Ve}+2 \cdot N_{Pu}$ = = 31	Anzahl der unbekannten Variablen = Anzahl der Gleichungen

Bild 3.41 zeigt die vollständige ungepackte Koeffizientenmatrix und die rechten Seiten des linearen Gleichungssystems für die Lösung dieses kleinen Beispiels für den ersten zu berechnenden Zeitpunkt.

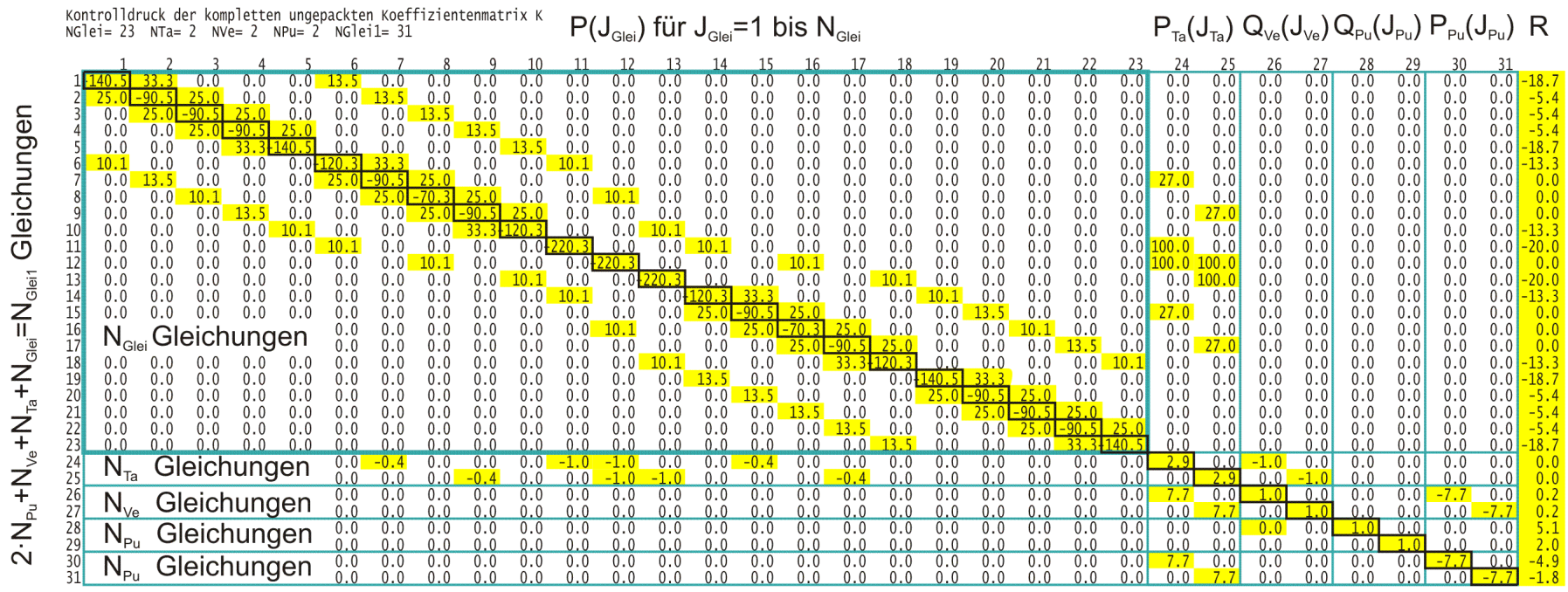


Bild 3.41: *Aufbau der ungepackten Koeffizienten Matrix K und der rechten Seite des linearen Gleichungssystems zur Berechnung des Schmierfilmdrucks und der primären Ergebnisdaten des peripheren Schmiersystems in der Hauptrechnung*

Die ersten N_{Glei} Gleichungen sind die Differenzengleichungen zur Approximation der Reynoldsschen Differentialgleichung zur Berechnung der Schmierfilmdruckverteilung. Sie machen den größten Teil des Gleichungssystems aus (siehe dazu Abschnitt 3.4.1). Die Erzeugung der Gleichungen und die Eintragung in die Koeffizientenmatrix und den Vektor der rechten Seiten erledigt die Routine KoMa2, wenn es Schmiertaschen gibt, oder KoMa1, wenn es keine Schmiertaschen gibt.

Die nächsten N_{Ta} Gleichungen bilden die Volumenbilanz in den Schmiertaschen ab. Für jede Schmiertasche wird genau eine Gleichung aufgestellt (siehe dazu Abschnitt 3.4.2). Die Erzeugung der Gleichungen und die Eintragung in die Koeffizientenmatrix und den Vektor der rechten Seiten erledigt die Routine KoMa3.

Die nächsten N_{Ve} Gleichungen repräsentieren die Gleichungen für die Kennlinien der Drosselgeräte in den Verbindungsleitungen. Sie stellen den Zusammenhang zwischen dem Pumpendruck P_{Pu}, dem Schmiertaschendruck P_{Ta} der angeschlossenen Schmiertasche und dem Schmiermittelstrom Q_{Ve} in der entsprechenden Verbindungsleitung dar.

Die nächsten N_{Pu} Gleichungen stellen die Aufteilung des Pumpenstroms Q_{Pu} auf die Ströme Q_{Ve} der angeschlossenen Verbindungsleitungen dar.

Die letzten N_{Pu} Gleichungen legen den Pumpendruck $P_{Pu}=P_{PuMax}$ fest, wenn die jeweilige Pumpe aktuell als druckgeregelte Pumpe betrieben wird, bzw. sie legen den Pumpenölstrom $Q_{Pu}=Q_{PuMax}$ fest, wenn die Pumpe aktuell als Konstantpumpe arbeitet. Zu den letzten N_{Pu} Gleichungen werden bei Bedarf noch ausgewählte Gleichungen aus dem Block der Gleichungen Nr. $N_{Glei}+N_{Ta}+1$ bis $N_{Glei}+N_{Ta}+N_{Ve}$ addiert (siehe dazu Abschnitt 3.4.3.2). Die letzten 3 Gleichungsblöcke enthalten meist recht triviale Gleichungen. Erzeugt werden sie durch die Routine KoMa4.

Die N_{Glei1} unbekannten Variablen sind passend zur Reihenfolge der Gleichungen sortiert. Zur besseren Anschaulichkeit wurden in Bild 3.41 die Blöcke der verschiedenen Gleichungen und die Blöcke der verschiedenen Unbekannten P, P_{Ta}, Q_{Ve}, Q_{Pu} und P_{Pu} durch eingetragene Linien getrennt. Außerdem wurden alle Koeffizienten, die von Null verschiedene Werte aufweisen, gelb hinterlegt. Die Koeffizienten der Hauptdiagonalen sind schwarz umrahmt. Es zeigt sich, dass schon bei diesem kleinen Beispiel die Matrix nur dünn belegt ist, d.h. dass die meisten Koeffizienten Null sind. Dieses Verhältnis wird bei üblicher Gitterteilung noch wesentlich extremer, weil sich die Anzahl der Koeffizienten pro Gleichung nur in den wenigen Gleichungen der 4 letzten Gleichungsblöcke erhöhen kann.

Die ungepackte Matrix ist noch einigermaßen übersichtlich. Sie wird in SIRIUS aber nur zu Testzwecken während der Programmentwicklung und zur Fehlersuche verwendet. Standardmäßig wird programmintern diese Matrix direkt in gepackter Form erzeugt. Hier werden die Aufgaben der Routinen KoMa1, KoMa2, KoMa3 und KoMa4 durch die Routinen KoMa1_pack, KomA2_pack, KoMa3_pack und KoMa4_pack ersetzt.

Für dieses einfache Beispiel soll jetzt auch die gepackte Matrix gezeigt werden.

```
Kontrolldruck der kompletten gepackten Koeffizientenmatrix
NGlei= 23  NTa= 2  NVe= 2  NPu= 2  NGlei1= 31

   M     Kv(M)   Kc(M)   JGlei Kp(JGlei) R(JGlei)
    1 -140.5285     1       1        0  -18.7371
    2   33.3333     2       2        3   -5.4038
    3   13.5095     6       3        7   -5.4038
    4   25.0000     1       4       11   -5.4038
    5  -90.5285     2       5       15  -18.7371
    6   25.0000     3       6       18  -13.3333
    7   13.5095     7       7       22    0.0000
    8   25.0000     2       8       27    0.0000
    9  -90.5285     3       9       32    0.0000
   10   25.0000     4      10       37  -13.3333
   11   13.5095     8      11       41  -20.0000
   12   25.0000     3      12       45    0.0000
   13  -90.5285     4      13       50  -20.0000
   14   25.0000     5      14       54  -13.3333
   15   13.5095     9      15       58    0.0000
   16   33.3333     4      16       63    0.0000
   17 -140.5285     5      17       68    0.0000
   18   13.5095    10      18       73  -13.3333
   19   10.1321     1      19       77  -18.7371
   20 -120.2642     6      20       80   -5.4038
   21   33.3333     7      21       84   -5.4038
   22   10.1321    11      22       88   -5.4038
   23   13.5095     2      23       92  -18.7371
   24   25.0000     6      24       95    0.0000
   25  -90.5285     7      25      101    0.0000
   26   25.0000     8      26      107    0.2262
   27   27.0190    24      27      110    0.2262
   28   10.1321     3      28      113    5.0930
   29   25.0000     7      29      114    2.0372
   30  -70.2642     8      30      115   -4.8667
   31   25.0000     9      31      117   -1.8110
   24   25.0000     6
   25  -90.5285     7
   26   25.0000     8
   27   27.0190    24
   28   10.1321     3
   29   25.0000     7
...
...
   93   13.5095    18
   94   33.3333    22
   95 -140.5285    23
   96   -0.4244     7
   97   -1.0472    11
   98   -1.0472    12
   99   -0.4244    15
  100    2.9432    24
  101   -1.0000    26
  102   -0.4244     9
  103   -1.0472    12
  104   -1.0472    13
  105   -0.4244    17
  106    2.9432    25
  107   -1.0000    27
  108    7.7280    24
  109    1.0000    26
  110   -7.7280    30
  111    7.7280    25
  112    1.0000    27
  113   -7.7280    31
  114    1.0000    28
  115    1.0000    29
  116    7.7280    24
  117   -7.7280    30
  118    7.7280    25
  119   -7.7280    31
```

Die gepackte Matrix besteht aus den 3 Vektoren Kv(M), Kc(M) und Kp(N_{Glei1}). M ist hier die Anzahl der Koeffizienten, die nicht gleich Null sind. In dem Vektor Kv sind alle Nicht-Null-Koeffizienten zeilenweise und lückenlos hintereinander abgelegt. Der Vektor Kc gibt zu jedem Nicht-Null-Koeffizienten an, in welcher Spalte er in der ungepackten Matrix stehen würde. Die Vektoren Kv und Kc enthalten jeweils M Elemente, wobei die Elemente des Vektors Kc ganzzahlig sind. Der Pionter-Vektor Kp(N_{Glei}+1) enthält N_{Glei}+1 ganze Zahlen. Die erste Zahl ist die Null. In fortlaufender Folge enthält er dann die Nummern des letzten Nicht-Null Koeffizienten jeder Zeile. Auf diese Weise werden zum Abspeichern jedes Koeffizienten und seiner Lage in der Matrix 3 Zahlen benötigt. Diese 3 Vektoren benötigen aber wesentlich weniger Speicherplatz als die komplette ungepackte Matrix K. Diese Angaben sind ausreichend, damit das GMRES-Verfahren und das Konditionierungsverfahren immer auf die richtigen Koeffizienten zugreifen können.

Die hier gezeigten Darstellungen der Matrizen kann sich auch der versierte Anwender vom Programm SIRIUS anzeigen lassen. Dazu sind im Quelltext der Routinen FilmDruck1 und FilmDruck2 unter der Kommentarzeile "Kontrolldruck der Koeffizientenmatrix" bereits 4 deaktivierte Befehlsblöcke abgelegt, mit denen jeweils die komplette gepackte Matrix (gemäß hier gezeigter Tabelle), die letzten Koeffizienten der gepackten Matrix, die komplette ungepackte Matrix (gemäß Bild 3.41) oder der rechte untere Quadrant der ungepackten Matrix angezeigt werden kann. Dazu ist die Datei "FilmDruck1.f" bzw. "Filmdruck2.f" mit dem Quelltext dieser Routinen zu öffnen und die Deaktivierung der Befehlszeilen aufzuheben, indem der Buchstabe "c" am Anfang der jeweiligen Befehlszeilen gelöscht wird. Anschließend ist die Datei wieder abzuspeichern und der Quelltext erneut zu kompilieren. Nach dem Neustart des Programms wird jetzt vor jeder Lösung des Gleichungssystems die Matrix bzw. ein Teil davon angezeigt. Das kann ein Hilfsmittel im Rahmen der Weiterentwicklung des Programms sein.

Die vollständigen Daten zu dem hier gezeigten Demonstrationsbeispiel findest Du in der Datei "Demo23.txt".

3.4.5 Das Lösungsverfahren zur Lösung des linearen Gleichungssystems

Ursprünglich wurde zur Lösung des Gleichungssystems ein Gaußsches Eliminationsverfahren verwendet (siehe Abschnitt 3.4.5.3). Da es trotz schneller werdender Rechner bei den aktuell üblichen Gitterteilungen noch recht langsam arbeitet, wurde ein für diese Aufgabe geeignetes schnelleres Verfahren gesucht. Wie bei der Simulation vieler technischer Sachverhalte ist die Koeffizientenmatrix des Gleichungssystems auch hier nur dünn besetzt. Dafür wurden parallel zur Entwicklung leistungsfähiger Computer und komplexer werdender Aufgaben eine Reihe leistungsfähiger Verfahren entwickelt, aus denen ein geeignetes ausgewählt wurde. Der Autor hat sich für das GMRES-Verfahren (Abschnitt 3.4.5.1) in Kombination mit einer Vorkonditionierung der Matrix mittels einer ILU-Zerlegung entschlossen (Abschnitt 3.4.5.2).

3.4.5.1 Das GMRES-Verfahren

Wikipedia gibt folgende Kurzbeschreibung des Verfahren: "*Das **GMRES-Verfahren** (für **Generalized minimal residual method**) ist ein iteratives numerisches Verfahren zur Lösung großer, dünnbesetzter linearer Gleichungssysteme. Das Verfahren ist aus der Klasse der Krylow-Unterraum-Verfahren und insbesondere auch für nicht-symmetrische Matrizen geeignet. In exakter Arithmetik, also wenn ohne Rundungsfehler gerechnet wird, liefert das Verfahren nach endlich vielen Schritten die exakte Lösung. Interessanter ist es jedoch als näherungsweises Verfahren, da es mit einer geeigneten Vorkonditionierung auch Gleichungssysteme mit Millionen Unbekannten in wenigen Iterationen mit befriedigender Genauigkeit lösen kann. Damit stellt es eine Art Black Box-Löser für dünnbesetzte lineare Gleichungssysteme dar. Es wurde 1986 von Yousef Saad und Martin H. Schultz entwickelt. ...*" [32] Hier wird auch bereits ein guter Überblick über den theoretischen Hintergrund und die Funktionsweise des Verfahrens gegeben.

Eine weitere informative Quelle zu diesem Verfahren ist das Lehrbuch von Meister [11]. Es liefert für dieses und andere Verfahren auch Quellcode in der Programmiersprache MATLAB als Vorlage.

Eine freie Quelle einer lauffähigen Routine des Verfahrens in der Programmiersprache FORTRAN wurde nicht gefunden, deshalb wurde vom Autor anhand der hier angegebenen Quellen das Verfahren in der Sprache FORTRAN77 neu programmiert. Es liegt vor in der Routine "GMRES_ILU_pack", die als kommentierter Quellcode in der gleichnamigen Textdatei "GMRES_ILU_pack.f" abgelegt ist. Es wurde eine Variante des Verfahrens mit Restart programmiert. Die Anzahl der Iterationen bis zu einem Restart ist programmintern mit dem Parameter **MaxIt=30** fest vorgegeben, da von diesem Wert die erforderliche Größe einiger Hilfsdatenfelder abhängt. Die Anzahl der Restarts, die das Programm ausführt, bevor es wegen Nicht-Konvergenz abbricht, ist durch den Parameter **Maxstart** angegeben. Standardmäßig ist der Wert mit 10 vorgegeben, so dass das Programm nach maximal 30·10=300 Iterationen wegen Nicht-Konvergenz abbricht. In SIRIUS kann über das Untermenü "Programminterne Parameter ändern" im Hauptmenü "Ende der Eingabe" der Parameter Maxstart vom Anwender ohne Eingriff in den Quellcode zeitweilig geändert werden.

Neben der Bedingung einer dünnbesetzten regulären Matrix, deren Hauptdiagonale voll besetzt sein muss, stellt das GMRES-Verfahren keine weiteren speziellen Anforderungen an das zu lösende Gleichungssystem. Die Routine "GMRES_ILU_pack" ist auch nicht speziell auf die Belange der Lagerberechnung zugeschnitten. Deshalb könnte die Routine "GMRES_ILU_pack" auch zur Lösung anderer Probleme in andere Programme implementiert werden. Es müssten lediglich die erforderlichen Feldgrößen, die in FORTRAN77 fest vorzugeben sind, im Quelltext entsprechend angepasst werden. Das erfolgt durch Festlegen eines geeigneten Wertes für den Parameter NG_{Max}.

Gegenüber dem Gaußschen Eliminationsverfahren brachte das GEMRES-Verfahren in Kombination mit der ILU-Zerlegung bei einer üblichen Gitterteilung der Schmierspaltfläche von ca. 300·20=6000 Gitterpunkten und damit ca. 36 Mio. Koeffizienten eine Verkürzung der Rechenzeit um den Faktor 1000. Bei direkter Erzeugung der gepackten Matrix wird auch Speicherplatz eingespart, weil keine ungepackte Matrix mehr erzeugt werden muss, deren Koeffizienten fast alle Null sind. Ein weiterer Vorteil ist, dass bei einer Vergrößerung der Anzahl der Gleichungen die Rechenzeit nur proportional ansteigt mit der Anzahl der Gleichungen, im Gegensatz zum Gauß-Verfahren, wo die Rechenzeit mit dem Faktor n^3 ansteigt.

3.4.5.2 Die Vorkonditionierung der Koeffizientenmatrix mit der ILU-Vorkonditionierung

Wikipedia gibt folgende kurze Erklärung für den Begriff: " *In der numerischen Mathematik bezeichnet **Vorkonditionierung** eine Technik, mittels derer ein Problem so umgeformt wird, dass die Lösung erhalten bleibt, sich jedoch für das gewählte numerische Lösungsverfahren positive Eigenschaften wie bessere Kondition oder schnellere Konvergenz ergeben.*

Die gebräuchlichste Form der Vorkonditionierung ist die lineare, bei der ein lineares Gleichungssystem $Ax = b$ äquivalent umgeformt wird. Diese Art der Vorkonditionierung findet insbesondere bei der Lösung des Gleichungssystems mittels Krylow-Unterraum-Verfahren Anwendung. ... " [38]

Die ILU-Zerlegung wird durch Wikipedia in folgender Weise kurz beschrieben:" *Als **ILU-Zerlegung** (von incomplete LU-Decomposition) oder unvollständige LU-Zerlegung bezeichnet man in der numerischen Mathematik die fehlerbehaftete Zerlegung einer Matrix* $A \in \mathbb{R}^{n \times n}$ *in das Produkt einer unteren Dreiecksmatrix L und einer oberen Dreiecksmatrix U*

$LU \approx A,$

bei der von den Zerlegungsmatrizen L und U nur die Einträge einer vorgegebenen Besetzungsstruktur berechnet werden. ...

Die ILU-Zerlegung wird erfolgreich als Vorkonditionierer zur Beschleunigung der iterativen Lösung großer dünnbesetzter linearer Gleichungssysteme $Ax = b$ *mittels Krylow-Unterraum-Verfahren eingesetzt. Es werden dabei keine Eigenschaften des eigentlichen Problems (meist die numerische Lösung einer partiellen Differentialgleichung) ausgenutzt. Damit ist sie nicht auf bestimmte Problemklassen beschränkt und hat Einzug in viele Bereiche der numerischen Simulation gefunden, beispielsweise in der numerischen Strömungsmechanik ist die Technik weit verbreitet. Zuerst erwähnt wurde das Verfahren 1960 von Richard S. Varga und Nikolai Iwanowitsch Bulejew (N. I. Buleev). Eine genauere Analyse wurde 1977 von J. A. Meijerink und van der Vorst veröffentlicht. ..."* [35]

Die Skripte zur Numerischen Mathematik von Lube [10] geben eine hilfreiche Darstellung für eine Umsetzung in ein Programm.

Das GMRES-Verfahren wird erst schnell durch eine geeignete Vorkonditionierung. In der Literatur wird dafür das Verfahren der ILU-Zerlegung empfohlen, weshalb sich der Autor auch für diese Vorkonditionierung entschieden hat. Eine freie Quelle einer lauffähigen Routine des Verfahrens der ILU-Zerlegung wurde nicht gefunden, deshalb wurde vom Autor anhand der hier angegebenen Quellen das Verfahren in der Sprache FORTRAN77 neu programmiert. Es liegt vor in der Routine "ILU_Zerlegung_pack", die als kommentierter Quellcode in der gleichnamigen Textdatei "ILU_Zerlegung_pack.f" abgelegt ist.

3.4.5.3 Das Gaußsche Eliminationsverfahren

Wikipedia gibt folgende Kurzbeschreibung des Verfahren: "*Das **Gaußsche Eliminationsverfahren** oder einfach **Gauß-Verfahren** (nach Carl Friedrich Gauß) ist ein Algorithmus aus den mathematischen Teilgebieten der linearen Algebra und der Numerik. Es ist ein wichtiges Verfahren zum Lösen von linearen Gleichungssystemen und beruht darauf, dass elementare Umformungen zwar das Gleichungssystem ändern, aber die Lösung erhalten. Dies erlaubt es, jedes eindeutig lösbare Gleichungssystem auf Stufenform zu bringen, an der die Lösung durch sukzessive Elimination der Unbekannten leicht ermittelt oder die Lösungsmenge abgelesen werden kann.*

Die Anzahl der benötigten Operationen ist bei einer $n \times n$*-Matrix* von der Größenordnung n^3. In seiner Grundform ist der Algorithmus anfällig für *Rundungsfehler*, aber mit kleinen Modifikationen (*Pivotisierung*) stellt er für allgemeine lineare Gleichungssysteme das Standardlösungsverfahren dar und ist Teil aller wesentlichen Programmbibliotheken für numerische lineare Algebra ..." [31]

Standardmäßig wird dieses Verfahren in SIRIUS nicht mehr verwendet, da es zu langsam arbeitet. Wegen der Schwierigkeiten, fehlerfreie Routinen zur Erstellung der gepackten Koeffizientenmatrix zu schreiben, wurde das Verfahren aber nicht vollständig aus dem Programm entfernt. Es kann zu Testzwecken wieder aktiviert werden. Siehe dazu Abschnitt 3.4.5.4.

Der Quellcode des Verfahrens liegt als Routine SIMQ in der gleichnamigen Textdatei "SIMQ.f" vor. Es stammt aus einer Programmbibliothek des Rechenzentrums der VVB Schiffbau Rostock, aus der es im Zeitraum zwischen 1974 und 1978 entnommen wurde. Der Autor der Routine ist nicht bekannt. Es stellt ein Gauß-Verfahren mit Pivotisierung dar.

3.4.5.4 Aktivierung der Berechnung mit ungepackter Matrix

Standardmäßig arbeitet das Programm SIRIUS zur Lösung des linearen Gleichungssystems innerhalb der Hauptrechnung mit einer gepackten Matrix, die durch die Routinen "KoMa1_pack" bzw. "Koma2_pack", "KoMa3_pack" und "KoMa4_pack" direkt erzeugt wird und durch das schnelle GMRES-Verfahren mit Vorkonditionierung (Routine "GMRES_ILU_pack") gelöst wird. Diese Routinen werden durch die übergeordneten Routinen "FilmDruck1" und "FilmDruck2" aufgerufen.

Um bei einer Überarbeitung oder Erweiterung des Programms die schwierige Programmierung der direkten Erzeugung gepackter Matrizen überprüfen zu können, kann die Berechnung über den langsameren Weg der Arbeit mit ungepackten Matrizen einfach aktiviert werden.

Dazu ist bereits je eine Variante der Routinen "FilmDruck1" und "FilmDruck2", die mit ungepackter Matrix arbeiten, vorbereitet und in den deaktivierten Quellcode-Dateien "FilmDruck1_unpack.f_" und "FilmDruck2_unpack.f_" abgelegt. In diesen Routinen wird zunächst mit den Unterroutinen "KoMa1", "KoMa2", "KoMa3" und "KoMa4" die ungepackte Matrix K erzeugt. Anschließend wird diese mit der Unterroutine "MatrixPacken" in eine gepackte Matrix umgewandelt und das Gleichungssystem mit der Routine "GMRES_ILU_pack" gelöst.

Um diese Berechnungsvariante zu aktivieren, sind die dazu erforderlichen Quellcode-Dateien zu aktivieren, indem die Dateiendungen ".f_" auf ".f" geändert werden. Das sind die Dateien

"FilmDruck1_unpack.f_",
"FilmDruck2_unpack.f_",
"KoMa1.f_",
"KoMa2.f_",
"KoMa3.f_",
"KoMa4.f_" und
"MatrixPacken.f_".

Außerdem **müssen** die beiden Quellcode-Dateien

"FilmDruck1.f" und
"FilmDruck2.f"

deaktiviert werden, indem die Dateierweiterungen ".f" in ".f_" geändert werden. Die dann auch nicht benötigten Quellcode-Dateien "KoMa1_pack.f" bzw. "Koma2_pack.f", "KoMa3_pack.f" und "KoMa4_pack.f" können ebenfalls deaktiviert werden, müssen aber nicht. Anschließend muss der gesamte Quellcode neu kompiliert (siehe Abschnitt 4.2.8) und danach die neu Programmdatei "SIRIUS.exe" gestartet werden.

ERLÄUTERUNG: *Der FORTRAN-Compiler erkennt nur Textdateien mit der Erweiterung ".f" als Quellcode-Dateien und bezieht nur diese in die Kompilierung ein. Das bewirkt der Eintrag "*.f" in der Batch-Datei "1-Start-CompilerG77.bat" Alle anderen Dateien werden ignoriert. So können Quellcode-Dateien leicht deaktiviert bzw. aktiviert werden, ohne sie in ein anders Verzeichnis verschieben zu müssen.*

TIPP: *Wenn die bereits kompilierte ausführbare Datei "SIRIUS.exe", die die Standardversion des Programms enthält, vorher umbenannt wird, dann wird sie durch die Neu-Kompilierung nicht überschrieben und kann weiter genutzt werden. So ist es möglich beide Programme parallel zu nutzen, was den Vergleich der Ergebnisse erleichtert.*

Es kann auch ganz auf die Arbeit mit einer nachträglich gepackten Matrix und der Lösung des Gleichungssystems mit dem GMRES-Verfahren verzichtet werden. Stattdessen wird dann die ungepackte Matrix direkt mit dem Gaußschen Eliminationsverfahren (Routine SIMQ) gelöst. Dazu sind zusätzlich in den inzwischen aktivierten Quellcode-Dateien "FilmDruck1_unpack.f", "FilmDruck2_unpack.f" folgende Befehlszeilen zu ändern:

```
...
      call MatrixPacken(K,NGlei1,Kv,Kc,Kp,M)
      call GMRES_ILU_pack(M,Kv,Kc,Kp,R,NGlei1,R1,Tol,MaxStart,Konvergenz)
...
c     call SIMQ(K,R,NGlei1,Konvergenz)
c     if(Konvergenz==1)then
c       write(*,*)
c       write(*,*)'FEHLERMELDUNG 401 in FilmDruck1_unpack:'
c       write(*,*)'  Die Gleichungen sind nicht linear unabhaenig.'
c       write(*,*)'  Die Berechnung wird abgebrochen.'
c       write(*,*)'Weiter mit ENTER:'
c       read(*,*)
c       Status=9
c       return
c     endif
c     do 99 jj=1,NGlei1
c99   R1(jj)=R(jj)
...
```

Bei allen gezeigten Befehlszeilen, die mit einem "c" in der ersten Spalte beginnen, ist dieses "c" zu entfernen. Bei allen gezeigten Befehlszeilen, wo dieses "c" fehlt, ist ein "c" einzufügen.

HINWEIS: *In FORTRAN77 werden Zeilen, die in der ersten Spalte mit dem Buchstaben "c" beginnen, vom Compiler als Kommentarzeilen interpretiert und beim Kompilieren ignoriert.*

HINWEIS: *Beachte, dass das Programm nach der Umstellung, wegen der begrenzten Größe der Matrix K auf maximal $N_{GleiMax} \cdot N_{GleiMax}$ Koeffizienten, jetzt nur mit einer geringeren maximalen Gitterteilung arbeiten kann, was wegen der geringeren Rechengeschwindigkeit sinnvoll ist.*

3.4.6 Korrekturroutinen zur Dämpfung von Instabilitäten der Druckberechnung

Die erweiterte Reynoldssche Differentialgleichung (Theo=2) ist nicht linear und kann deshalb mit dem Differenzenverfahren nur iterativ gelöst werden. Bei der Berechnung der Druckverteilung können deshalb Instabilitäten entstehen, die sich aufschaukeln und schließlich zum Abbruch der Berechnung führen (siehe ausführliche Erläuterung dazu im Abschnitt 4.9.2.1). Prinzipiell lassen sich diese Probleme durch feinere Gitterteilungen ΔX und/oder kleinere Zeitschrittweiten ΔT beseitigen. Das kann aber zu erheblich höherem Berechnungsaufwand führen und evtl. auch die Kapazität des Berechnungsprogramms oder die Geduld des Anwenders überfordern.

Mit Hilfe empirischer Untersuchungen und einiger Überlegungen wurden Wege gefunden, diese Instabilitäten, wenn auch nicht vollständig zu beseitigen, so doch erheblich zu dämpfen, so dass mit relativ groben Gitterteilungen N_X und Zeitschrittweiten N_T bereits gute Ergebnisse erzielt werden und so Berechnungszeit gespart wird. Das wird durch die Korrekturroutinen "Pglatt", "Fuell1", "KleiDru4" und "Pkorr6" realisiert. Da sie sich bewährt haben, sind sie fest in das Programm eingefügt.

Bild 3.45 zeigt die Flüssigkeitsverteilung im Schmierspalt zum Zeitpunkt J_T=10 für ein schwellend belastetes Lager. Noch anschaulicher ist die dazugehörige Animation von 2 Lastzyklen. Die Daten zu diesem Demonstrationsbeispiel sind in der Datei "Demo24-1.txt" abgelegt. Die Animation und das Bild zeigen, dass die Berechnung stabil abläuft.

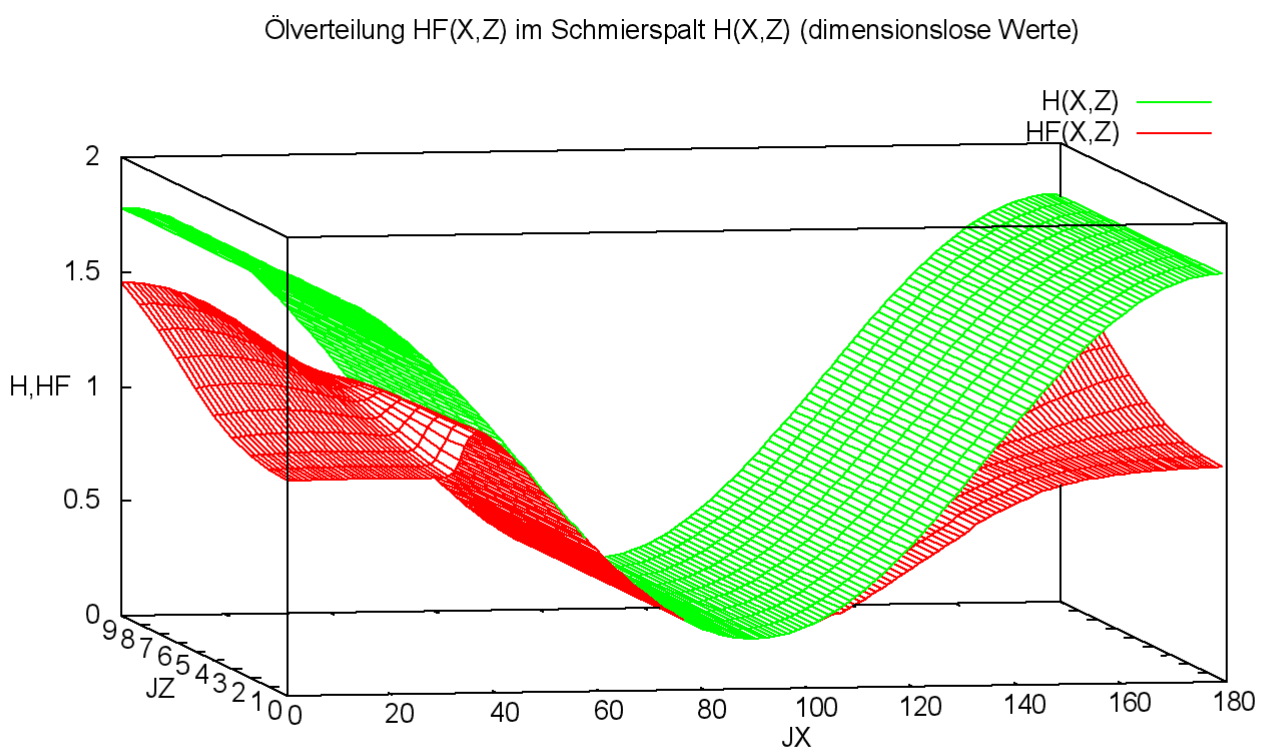


Bild 3.45: *Stabiles Berechnungsergebnis des Programms SIRIUS mit den Korrekturroutinen "Pglatt" und "Pkorr6" (Bilddatei: Demo24-1-3d-Abw-H-HF-JT=10.png) (Animation: Demo24-1-3d-Abw-H-HF.wmv)*

Bild 3.46 zeigt nun die Flüssigkeitsverteilung für das gleiche Beispiel für den gleichen Zeitpunkt der Berechnung. Hier wurden aber zuvor die Routinen "Pglatt" und "Pkorr6" im Programm deaktiviert. Der gezeigte Zeitpunkt ist der Zeitpunkt, zu dem erstmals eine Instabilität auftritt. Bild 3.47 zeigt dann den letzten Zeitpunkt der Berechnung bevor das Programm mit einer Fehlermeldung abbricht. Es zeigt, dass sich die Instabilität fast über den gesamten Schmierspalt ausgebreitet hat.

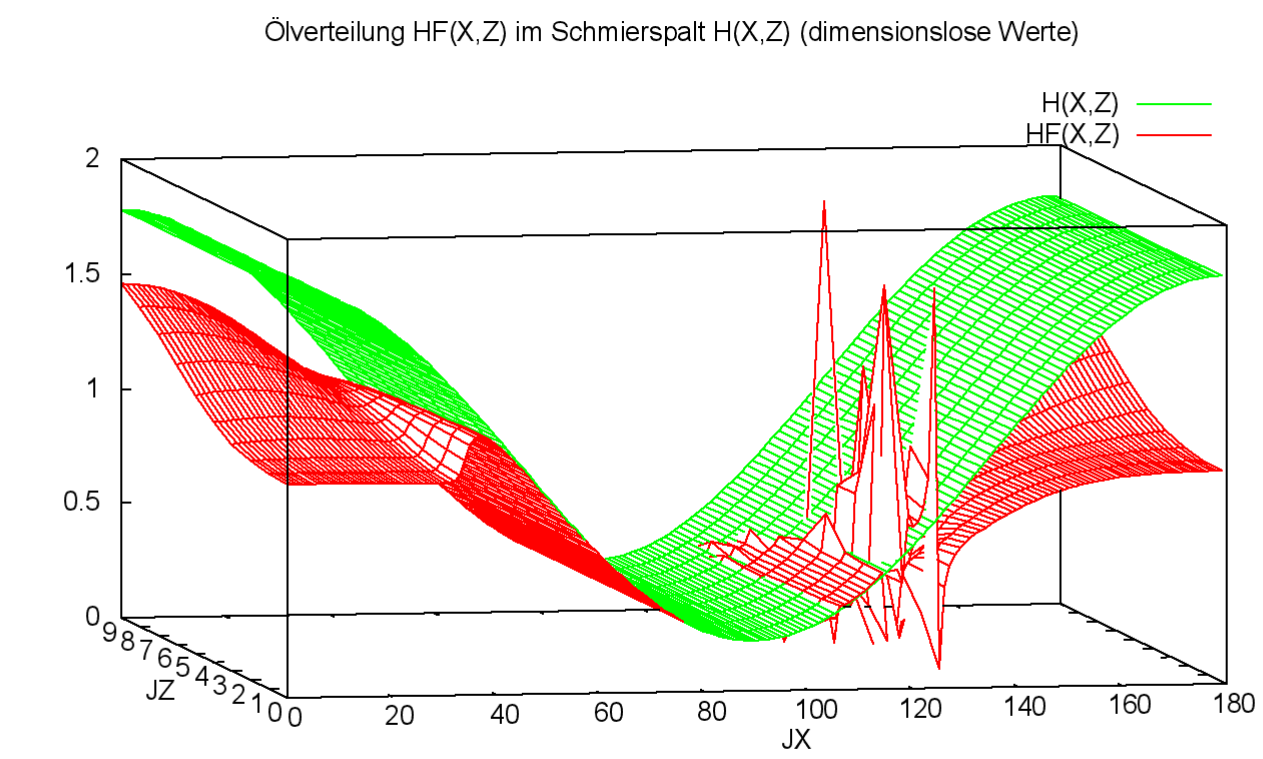


Bild 3.46: *Erstes instabiles Berechnungsergebnis des Programms SIRIUS ohne Korrekturroutinen "Pglatt" und "Pkorr6" (Bilddatei: Demo24-2-3d-Abw-H-HF-JT=10.png) (Animation: Demo24-2-3d-Abw-H-HF.wmv)*

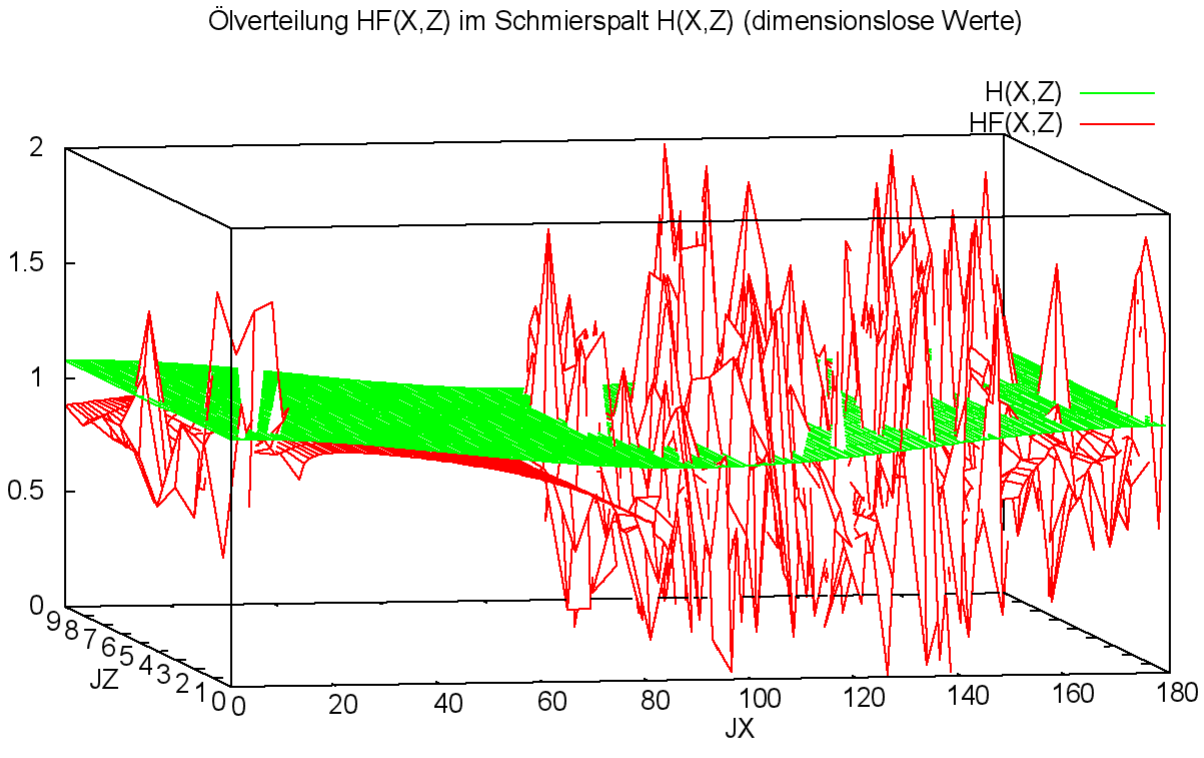


Bild 3.47: *Letztes instabiles Berechnungsergebnis des Programms SIRIUS ohne Korrekturroutinen "Pglatt" und "Pkorr6" vor dem Abbruch der Berechnung (Bilddatei: Demo24-2-3d-Abw-H-HF-JT=19.png)*

Die Ergebnisse dieser Berechnung sind in der Datei "Demo24-2.txt" abgelegt. Du kannst sie aber nur nachrechnen, wenn Du vorher im Quellcode der Routine "FilmDruck2" die Routinen "Pglatt" und "PKorr6" deaktivierst. Sonst würden wieder die Ergebnisse aus "Demo24-1.txt" erscheinen.

Wie gesagt, könnte man prinzipiell auch ohne die Korrekturroutinen auskommen. Dazu muss man nur die Gitterweite ΔX und die Zeitschrittweite ΔT ausreichend verkleinern. Im gezeigten Demonstrationsbeispiel musste dazu die Gitterweite ΔX halbiert und die Zeitschrittweite ΔT durch 3 geteilt werden. Das entspricht etwa einer 6-fachen Rechenzeit.

Das Bild 3.48 zeigt das wieder stabile Berechnungsergebnis ohne Korrekturroutinen aber mit aufwendigerer Berechnung. Die Ergebnisse sind fast identisch mit den Ergebnissen des Bildes 3.45.

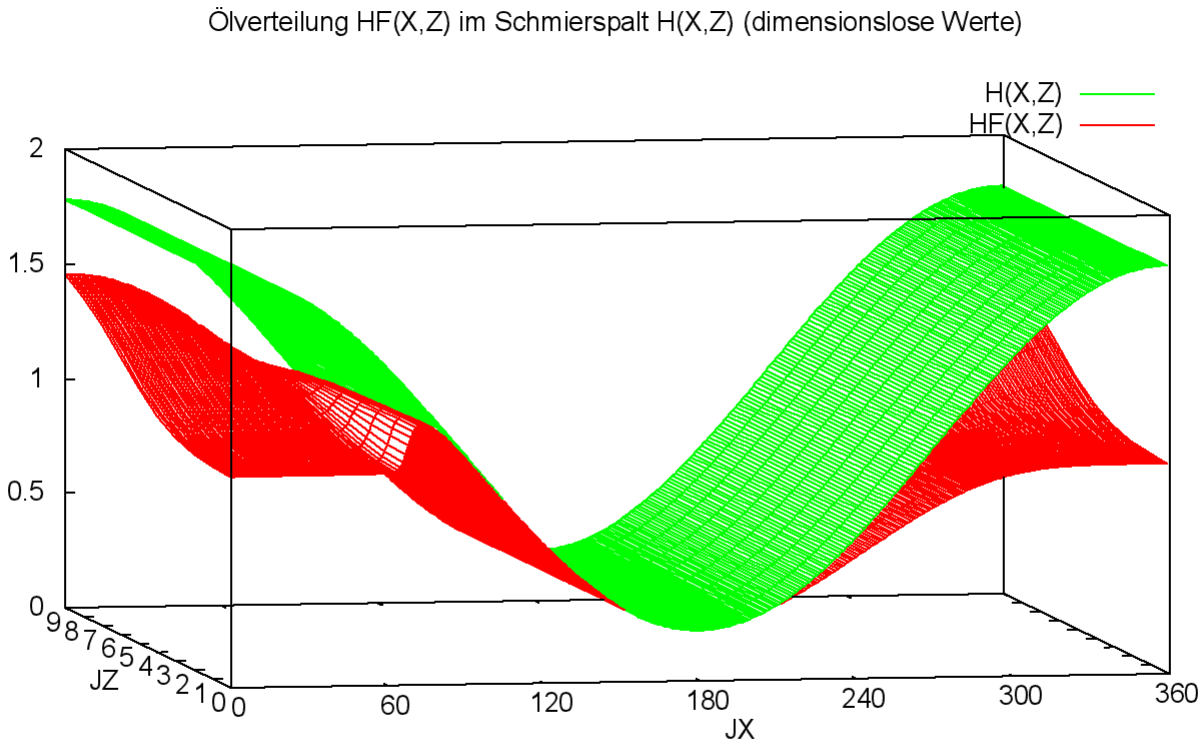


Bild 3.48: *Stabiles Berechnungsergebnis des Programms SIRIUS ohne Korrekturroutinen "Pglatt" und "Pkorr6" (Bilddatei: Demo24-3-3d-Abw-H-HF-JT=37.png) (Animation: Demo24-3-3d-Abw-H-HF.wmv)*

3.4.6.1 Glättung des Druckverlaufs im Gebiet der Kavitation mit der Routine "Pglatt"

Im folgenden Beispiel (Bild 3.49)wurden wieder die Eingabedaten des Demonstrationsbeispiels Demo24-1 verwendet. Es wurde dieses Mal im Quelltext nur die Routine "Pglatt" deaktiviert. Hier entsteht eine Instabilität im Bereich des Schmiermittelrückstaus vor dem Druckberganfang, der zu diesem welligen Verlauf der Schmiermittelverteilung führt. Im hier gezeigten Beispiel (Demo24-4.txt) tritt um den Zeitpunkt J_T=32 nur eine kurze lokale Instabilität am Druckberganfang auf, die dann wieder abklingt. Unter ungünstigeren Bedingungen kann sie sich aber auch leicht aufschaukeln.

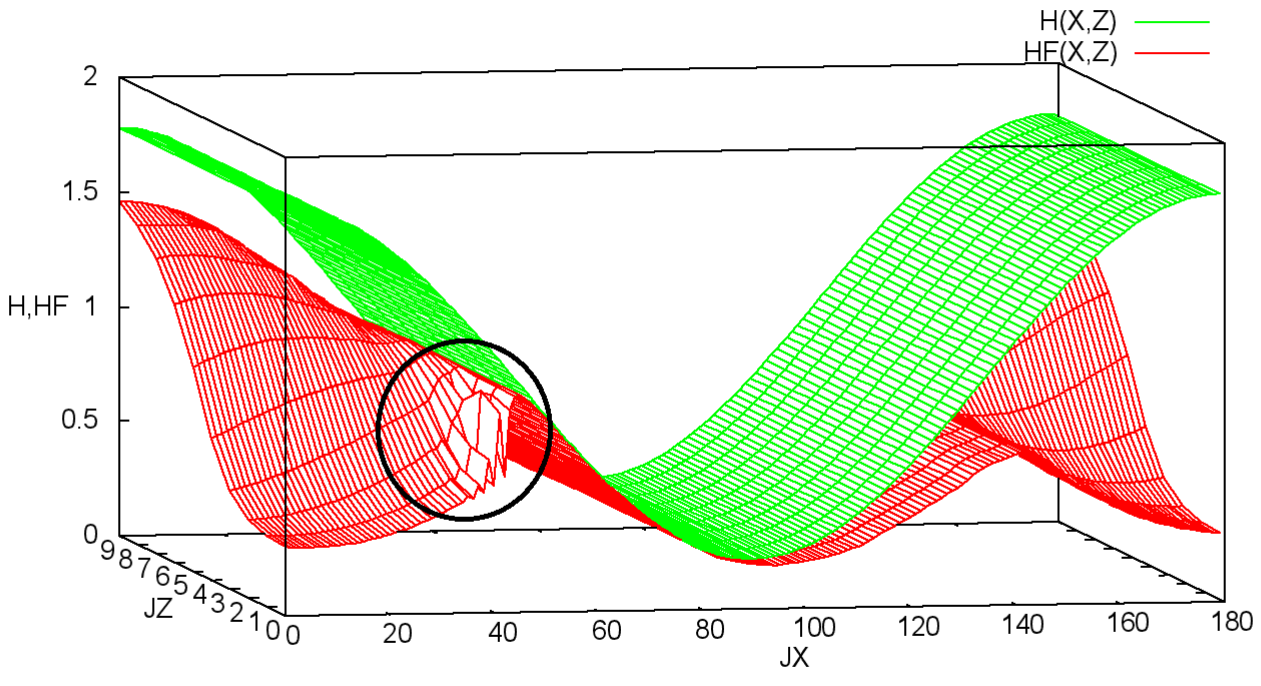


Bild 3.49: *Leicht instabiles Berechnungsergebnis des Programms SIRIUS ohne Korrekturroutine "Pglatt" (Animation: Demo24-4-3d-Abw-H-HF.wmv)*

Es wurden empirische Untersuchungen mit verschiedenen Glättungsfunktionen durchgeführt, um den Aufbau dieser Schwingungen zu unterdrücken oder mindestens zu dämpfen. Die besten Ergebnisse wurden mit nachfolgender Glättungsfunktion erreicht

$$P_{geglättet}(J_z,J_x)=\begin{pmatrix} P(J_z-1,J_X-1) & +2\cdot P(J_z-1,J_X) & +P(J_z-1,J_X+1)+ \\ +4\cdot P(J_z,J_X-1) & +8\cdot P(J_z,J_X) & +4\cdot P(J_z,J_X+1)+ \\ +P(J_z+1,J_X-1) & +2\cdot P(J_z-1,J_X) & +P(J_z+1,J_X+1) \end{pmatrix} / 24 \qquad (3.139)$$

Mit der Routine "Pglatt" wurde diese Glättungsfunktion in das Programm eingebaut. Die Glättungsroutine wird nach jeder Berechnung des Druckverlaufs mit der erweiterten Reynoldsschen Differentialgleichung (Theo=2) von der übergeordneten Routine "FilmDruck2" aufgerufen. Die Glättung wird aber nur im Bereich niedriger Drücke ausgeführt. Als Obergrenze der Korrektur wird das Druckniveau des kleinsten Drucks am Lagerrand P_{Rand1} bzw. P_{Rand2} angenommen.

Mit dieser Druckkorrektur können einige Instabilitäten im Anfangsstadium gut gedämpft werden, so dass diese sich oft nicht aufschaukeln. Diese Instabilitäten lassen sich aber nicht immer verhindern. Siehe dazu auch die Abschnitte 4.9.2.1 und 4.9.2.2.

HINWEIS: *Die Ergebnisse des gezeigten Beispiels sind in der Datei "Demo24-4.txt" abgelegt. Du kannst die Ergebnisse aufrufen und ohne Neuberechnung ansehen. Wenn Du aber versucht, die Ergebnisse nachzurechnen, werden die Ergebnisse von Demo24-1.txt erscheinen, weil im normalen Programm die Funktion geglättet wird und diese Instabilität deshalb nicht auftritt. Willst Du die gezeigte Instabilität simulieren, musst Du zuvor in den Quellcode eingreifen, in der Routine "FilmDruck2" den Aufruf der Routine "Pglatt" deaktivieren und das gesamte Programm neu kompilieren.*

3.4.6.2 Druckkorrektur durch Berechnung der minimalen Schmiermittelfüllung im Gebiet der Kavitation

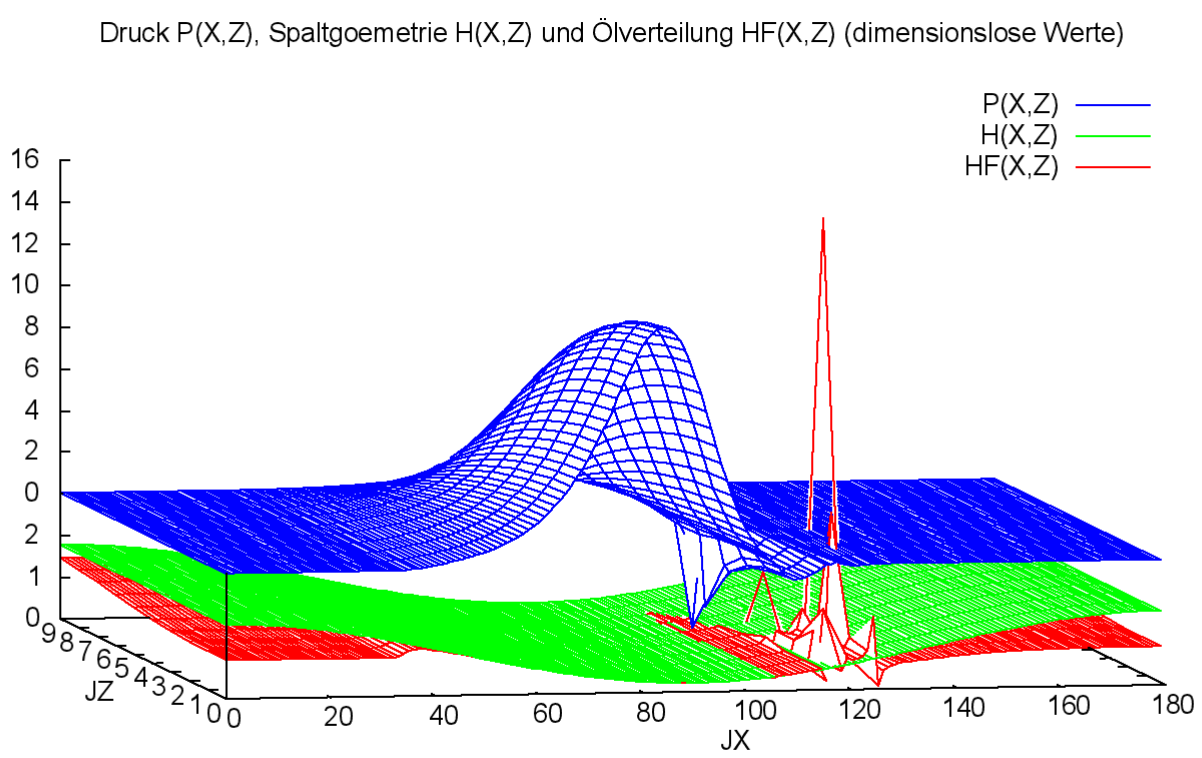


Bild 3.50: *Erstes instabiles Berechnungsergebnis des Programms SIRIUS ohne Korrekturroutinen "Pglatt" und "Pkorr6" (Bilddatei: Demo24-2-3d-Abw-P-H-HF-JT=10.png) (Animation: Demo24-2-3d-Abw-P-H-HF.wmv)*

Plötzliche Druckschwankungen, z.B. infolge einer starken Lastrichtungsänderung, können für die iterative Berechnung des Druckverlaufs im Schmierspalt nach der erweiterten Reynoldsschen Gleichung (Theo=2) ebenfalls zum Problem werden. Wenn ein Druckberg oder ein Teil davon plötzlich zusammenbricht und zum Kavitationsgebiet wird, kommt es vor, dass das iterative Lösungsverfahren negative Drücke berechnet, obwohl diese hier nicht zulässig sind. Dieser Effekt ist z.B. in Bild 3.50 zu beobachten. Hier wurde das Demonstrationsbeispiel "Demo24-2" ohne die Korrekturprogramme "Pglatt" und "Pkorr6" berechnet.

Um diesem Problem zu begegnen, wurde neben dem Glätten des Druckverlaufs ein weiteres Korrekturverfahren entwickelt, dass sich bewährt hat. Diesem Verfahren liegt folgende theoretische Überlegung zugrunde: Im Kavitationsgebiet, wo nur noch geringe Drücke und damit nur geringe Druckschwankungen herrschen, wird der Schmiermitteltransport fast ausschließlich durch Scherung angetrieben. D.h., dass die mittlere Schmiermittelgeschwindigkeit der halben Drehgeschwindigkeit der Welle entspricht. Wenn also die Schmierflüssigkeitsverteilung $HF(Z,X,J_T\text{-}1)$ aus dem vorhergehenden Zeitpunkt $J_T\text{-}1$ bekannt ist, kann man die Schmiermittelverteilung $HF(Z,X,J_T)$ für den aktuellen Zeitpunkt J_T gut abschätzen, indem man die bekannte vorhergehende Schmiermittelverteilung um den Weg $\Omega\cdot\Delta T/2$ in Drehrichtung der Welle verschiebt. Da es zu der erweiterten Reynoldsschen Gleichung einen direkten Zusammenhang zwischen dem Schmierfilmdruck und dem örtlichen Füllungsgrad und damit auch zu der örtlichen Flüssigkeitsmenge HF=H·F im Schmierspalt gibt, kann man aus der verschobenen Schmiermittelverteilung den Schmiermitteldruck Pk(Z,X) im Kavitationsgebiet abschätzen mit

$$(3.140) \qquad Pk(Z,X) = \frac{C \cdot HF(Z,X)}{H(Z,X) - HF(Z,X)}$$

Da die Drücke rund um das Kavitationsgebiet höher sind, strömt zusätzlich zur reinen Scherströmung Schmiermittel in das Kavitationsgebiet. D.h., dass der exakt berechnete Schmiermitteldruck P(Z,X) größer, aber nicht kleiner sein kann, als der mit Pk(Z,X) abgeschätzte Schmierfilmdruck. D.h. außerdem, dass Pk(Z,X) als Untergrenze des möglichen Schmierfilmdruckes P(Z,X) angesehen werden kann und jeder berechnete Druck P(Z,X), der kleiner ist als Pk(X,Z), ist fehlerhaft. Damit ergibt sich eine plausible Korrekturmöglichkeit für die iterativ berechneten Drücke im Kavitationsgebiet: Alle Drücke P(Z,X)<Pk(Z,X) werden auf PK(Z,X) hochgesetzt. Damit entsprechen diese Werte vielleicht nicht ganz der exakten Lösung. Es hat sich aber gezeigt, dass sie eine noch recht gute Näherung darstellen. Vor allem wird damit verhindert, dass Werte kleiner oder gleich Null in die Berechnung des nächsten Zeitschrittes eingehen, was zu unsinnigen Ergebnissen oder zum Abbruch des Programms führen würde.

Die hier beschriebene Korrektur wird in 3 Teilschritten durch die 3 Routinen "Fuell1", KleiDru4" und "Pkorr6" ausgeführt. Die Routine "Fuell1" berechnet bereits zum vorhergehenden Zeitpunkt $J_T\text{-}1$ aus der Druckverteilung die Schmierflüssigkeitsverteilung HF und verschiebt diese um den Betrag $\Omega\cdot\Delta T/2$ in Wellendrehrichtung. Die Routine "KleiDru4" berechnet aus dem Feld der abgeschätzten örtlichen Verteilung HF der Schmierflüssigkeit den KorrekturDruck Pk. Die Routine "Pkorr6" vergleicht die aktuell berechnete Druckverteilung P(Z,X) mit den abgeschätzten kleinsten Drücken Pk(Z,X) und korrigiert gegebenenfalls damit P(Z,X).

Die Korrektur mit Pk erfolgt erst nach dem Glätten der Druckverteilung durch die Routine "Pglatt". Siehe vorhergehenden Abschnitt 3.4.6.1.

Diese Korrekturroutinen haben sich bewährt und sind im Programm fest eingebaut. Sie arbeiten unsichtbar und ohne Zutun des Programmanwenders und können nur durch Eingriff in den Quelltext außer Kraft gesetzt werden. Wenn Du das Programm ohne diese Korrektur testen willst, brauchst Du nur im Quellcode der Routine "FilmDruck2" den Aufruf der Routine "Pkorr6" deaktivieren. Nach dem Kompilieren des Quelltextes arbeitet das Programm ohne diese Korrektur.

Diese Korrekturen werden nur bei der Verwendung der erweiterten Reynoldsschen Differentialgleichung (Theo=2) angewendet. Die klassische Reynoldssche Differentialgleichung (Theo=1) benötigt diese Korrektur nicht. Hier werden berechnete negative Drücke entsprechend der "Gümbelschen Randbedingung" mit der Routine "Pkorr1" hochgesetzt.

3.4.7 Die iterative Berechnung der Verlagerungsbahn aus einem vorgegebenen Belastungsverlauf

Mit Hilfe der Reynoldsschen Differentialgleichung kann man aus einer vorgegebenen Spaltgeometrie den Druckverlauf im Schmierspalt berechnen und durch Integration der Schmierspaltdrücke den resultierenden Vektor der Lagerbelastung $[F_1,F_2]$.

In der Praxis ist die Problemstellung aber umgekehrt. Es ist eine stationäre Lagerbelastung oder ein zeitlicher Verlauf der Lagerbelastung gegeben. Infolge des Vektors der Lagerbelastung $[F_1(T),F_2(T)]$ verlagert sich die Welle innerhalb des Lagerspielraums, bis der dadurch entstehende Schmierfilmdruck mit der Lagerbelastung im Gleichgewicht steht. Es ist also der Vektor der Wellenverlagerung [E1(T),E2(T)] zu ermitteln.

Deshalb musste eine algorithmierbare Suchstrategie entwickelt werden und als ein weiterer Iterationsprozess in das Berechnungsprogramm implementiert werden. Brökel [2] entwickelte eine entsprechende Suchstrategie und implementierte sie erstmals in das Programm SIRIUS.

Die nachfolgenden Abschnitte 3.4.7.1 bis 3.4.7.6 beschreiben die weiterentwickelte Lösung dieser Aufgabe ausführlich.

3.4.7.1 Formulierung des Problems der Verlagerungsbahnberechnung und ihr prinzipieller Ablauf

Formal kann man den Zusammenhang zwischen den Komponenten der Lagerbelastung F_1 und F_2 und der Spaltgeometrie E_1 und E_2 als die Funktionen funktion1 und funktion2 formulieren.

$$(3.141) \qquad F_1(J_T) = funktion1\left(E_1(J_T),E_2(J_T),\frac{\partial E_1(J_T)}{\partial T},\frac{\partial E_2(J_T)}{\partial T},E_1(J_t-1),E_2(J_T),\Delta T,\ldots\right)$$

$$(3.142) \qquad F_2(J_T) = funktion2\left(E_1(J_T),E_2(J_T),\frac{\partial E_1(J_T)}{\partial T},\frac{\partial E_2(J_T)}{\partial T},E_1(J_t-1),E_2(J_T),\Delta T,\ldots\right)$$

Es sind hier zunächst die 4 Variablen $E_1(J_T),E_2(J_T),\frac{\partial E_1(J_T)}{\partial T},\frac{\partial E_2(J_T)}{\partial T}$ unbekannt. Die beiden Ableitungen kann man aber auch näherungsweise durch die beiden unbekannten $E_1(J_T)$, $E_2(J_T)$ und die beiden bereits bekannten Werte $E_1(J_T\text{-}1),E_2(J_T\text{-}1)$ ausdrücken.

$$(3.143) \qquad \frac{\partial E_1(J_T)}{\partial T} \approx \frac{E_1(J_T)-E_1(J_T-1)}{\Delta T}$$

und

$$(3.144) \qquad \frac{\partial E_2(J_T)}{\partial T} \approx \frac{E_2(J_T)-E_2(J_T-1)}{\Delta T}$$

So lassen sich die beiden Funktionen um schreiben auf ein System mit nur noch 2 unbekannten Variablen, wobei zur besseren Übersicht alle bereits bekannten Parameter nicht mitgeschrieben werden.

$$(3.145) \qquad F_1(J_T) = func1\left(E_1(J_T),E_2(J_T)\right)$$

$$(3.146) \qquad F_2(J_T) = \text{func}2\left(E_1(J_T), E_2(J_T)\right)$$

Damit ist ein nichtlineares Gleichungssystem mit zwei Gleichungen und zwei Unbekannten gegeben.

Für diese beiden Funktionen gibt es keinen analytischen Ausdruck. Es gibt lediglich ein Berechnungsverfahren gemäß den Abschnitten 3.4.1 bis 3.4.5, das für jeden Punkt des Definitionsbereichs die Funktionswerte berechnen kann.

Es ist nun in folgender Weise vorzugehen:

Zunächst sind für die gesuchten Werte $E_1(J_T)$ und $E_2(J_T)$ zwei Anfangswerte $E_{1,0}$ und $E_{2,0}$ festzulegen. Geeignete Anfangswerte können aus dem vorangegangen Verlagerungsverlauf extrapolierte Werte.

Für diese Anfangswerte können die Funktionswert $F_{1,0}$ und $F_{2,0}$ berechnet werden, die aber noch nicht mit $F_1(J_T)$ und $F_2(J_T)$ übereinstimmen.

$$(3.147) \qquad F_1(J_T) \neq F_{1,0} = \text{func}1\left(E_{1,0}, E_{2,0}\right)$$

$$(3.148) \qquad F_2(J_T) \neq F_{2,0} = \text{func}2\left(E_{1,0}, E_{2,0}\right)$$

Nun sind verbesserte Werte E_{13} und E_{23} zu suchen, deren Funktionswerte F_{13} und F_{23} sich der vorgegebenen Lagerbelastung $F_1(J_T)$ und $F_2(J_T)$ weiter annähern

$$(3.149) \qquad F_{1,3} = \text{func}1\left(E_{1,3}, E_{2,3}\right)$$

$$(3.150) \qquad F_{2,3} = \text{func}2\left(E_{1,3}, E_{2,3}\right)$$

Als nächstes ist anhand vorgegebener Genauigkeitskriterien zu überprüfen, ob die Genauigkeit ausreichend ist und so die Iteration erfolgreich beendet werden kann. Siehe dazu Abschnitt 3.4.7.6.

Ist das nicht der Fall, sind die verbesserten Werte als neue Anfangsnäherungswerte zu setzen

$$(3.151) \qquad E_{1,3} \Rightarrow E_{1,0}$$

$$E_{2,3} \Rightarrow E_{2,0}$$

$$F_{1,3} \Rightarrow F_{1,0}$$

$$F_{2,3} \Rightarrow F_{2,0}$$

und ein erneuter Iterationszyklus auszuführen. Das ist zu wiederholen, bis die Genauigkeitskriterien erfüllt sind.

Da nicht jede Iteration zu einer Lösung konvergieren muss, wird zur Sicherheit die maximale Anzahl der zugelassenen Iterationszyklen auf einen Wert NJ begrenzt. Wenn bis dahin keine ausreichende Genauigkeit erzielt wurde, wird die Iteration abgebrochen und eine entsprechende Fehlermeldung ausgegeben.

Die Flussdiagramme in den Bildern 3.05 und 3.06 deuten diese Iterationsschleife (rote Schleife) in sehr vereinfachter Form an.

3.4.7.2 Geometrische Darstellung der Suchstrategie

Die Suchstrategie wird hier beschrieben, mit welcher aus den Anfangsnäherungswerten $E_{1,0}(J_T)$, $E_{2,0}(J_T)$ die verbesserten Näherungswerte $E_{1,3}(J_T)$, $E_{2,3}(J_T)$ berechnet werden.

Kern des Problems ist die Formulierung einer zweckmäßigen Suchstrategie, die mit wenigen Berechnungen der Funktionen func1($E_{1...}$,$E_{2...}$) und func2($E_{1...}$,$E_{2...}$) auskommt, da sich hinter jeder Berechnung jeweils eine komplette Berechnung des Druckverlaufs P(Z,X) im Schmierspalt verbirgt.

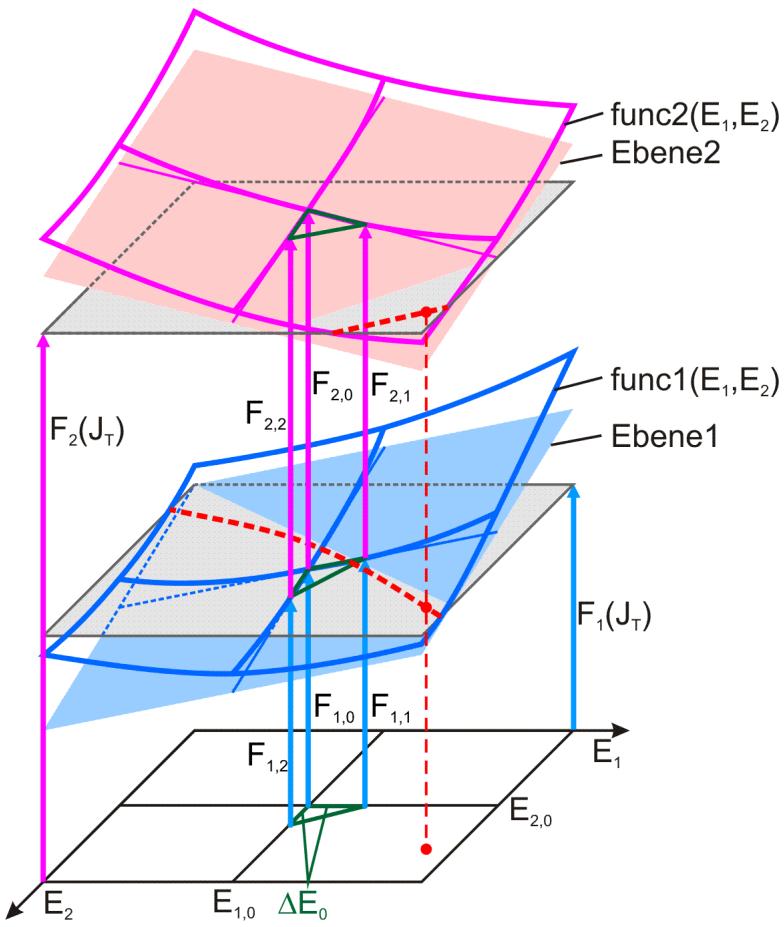


Bild 3.51: *Skizzen der Funktionen func1 und func2 in der Umgebung der Anfangsnäherung $E_{1,0}$, $E_{2,0}$*

Bild 3.51 skizziert die Funktionen func1 und func2 in der Umgebung der Anfangsnäherung [$E_{1,0}$,$E_{2,0}$] und dazu die Höhenebenen(in grauer Farbe) mit den Höhen $F_1(J_T)$ und $F_2(J_T)$. Die exakte Lösung liegt dort, wo die Funktion func1 die Höhenebene $F_1(J_T)$ schneidet und gleichzeitig die Funktion func2 die Höhenebene $F_2(J_T)$ schneidet, denn das ist die Lösung des nichtlinearen Gleichungssystems (3.145), (3.146). Sowohl die Schnittlinien (gestrichelte rote Linien) der beiden Funktionen mit der entsprechenden Höhenebene, als auch der Kreuzungspunkt der Schnittlinien sind nicht direkt zu ermitteln. Deshalb werden beide Funktionen in der Umgebung der Anfangs-näherungslösung [$E_{1,0}$;$E_{2,0}$] durch die Ebenen 1 und 2 approximiert. Dazu werden die Funktionswerte der Funktionen func1 und func2 an je 2 weiteren Stellen ermittelt.

$$F_{1,1} = \text{func1}\left(E_{1,1} = E_{1,0} + \Delta E_0; E_{2,0}\right) \tag{3.152}$$

$$F_{1,2} = \text{func1}\left(E_{1,0}; E_{2,2} = E_{2,0} + \Delta E_0\right) \tag{3.153}$$

$$F_{2,1} = \text{func2}\left(E_{1,1} = E_{1,0} + \Delta E_0; E_{2,0}\right) \tag{3.154}$$

$$F_{2,2} = \text{func2}\left(E_{1,0}; E_{2,2} = E_{2,0} + \Delta E_0\right) \tag{3.155}$$

Mit den jeweils drei Stützstellen $F_{1,0}$; $F_{1,1}$; $F_{1,2}$ und $F_{2,0}$; $F_{2,1}$; $F_{2,2}$ können die Ebenen 1 und 2 aufgespannt werden. Je kleiner ΔE_0 gewählt wird, umso exakter bilden die beiden Ebenen die Tangentialebenen der beiden Funktionen im Punkt $[E_{1,0};E_{2,0}]$ ab. Mit Hilfe der tangentialen Ebenen 1 und 2 und den beiden Höhenebenen kann nun, sowohl grafisch als auch rechentechnisch, eine verbesserte Näherungslösung für das nichtlinearen Gleichungssystems (3.145), (3.146) gefunden werden.

Bild 3.52 zeigt die grafische Lösung des Problems. Die verbesserte Näherungslösung $[E_{1,3};E_{2,3}]$ ist dann

$$F_1(J_T) = F_{1,3} = \text{Ebene1}\left(E_{1,3}; E_{2,3}\right) \tag{3.156}$$

$$F_2(J_T) = F_{2,3} = \text{Ebene2}\left(E_{1,3}; E_{2,3}\right) \tag{3.157}$$

mit

$$E_{1,3} = E_{1,0} + \Delta E_1 \tag{3.158}$$

und

$$E_{2,3} = E_{2,0} + \Delta E_2 \tag{3.159}$$

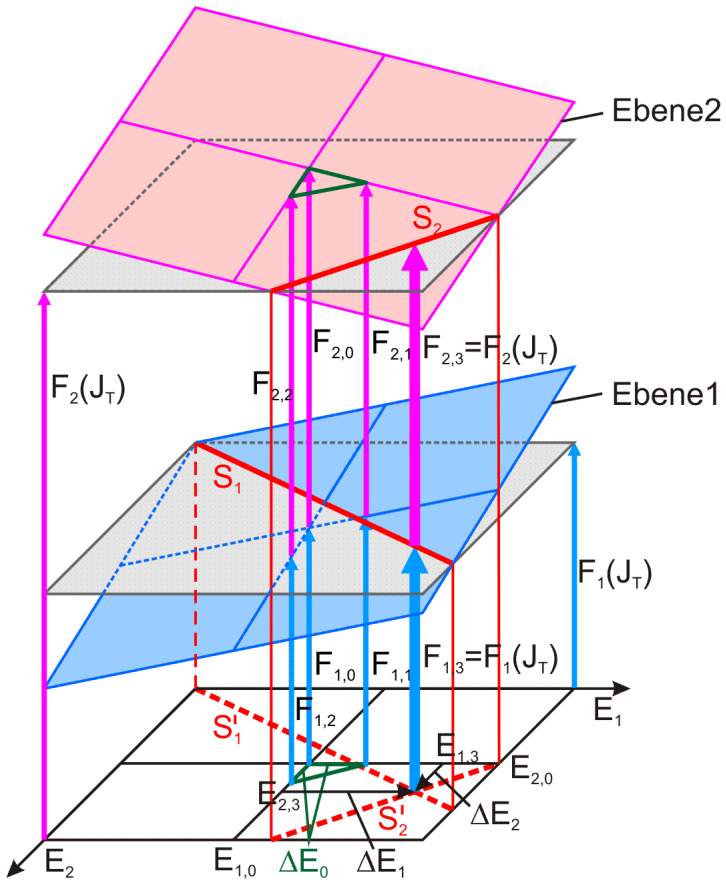


Bild 3.52: *Darstellung der grafischen Lösung der Ermittlung eines verbesserten Wellenverlagerungspunktes $[E_1,E_2]$*

Die Formeln zur nummerischen Lösung des Problems werden im folgenden Abschnitt angegeben.

3.4.7.3 Numerische Darstellung der Suchstrategie

Aus der grafischen Lösung können nun leicht die Formeln für die numerische Lösung abgeleitet werden. Die Gleichungen für die tangentiale Ebene sind gegeben durch

Ebene1:

$$F_1(E_1;E_2) = a_{11} \cdot E_1 + a_{12} \cdot E_2 + a_{13} \tag{3.160}$$

mit

$$a_{11} = \frac{F_{1,1} - F_{1,0}}{\Delta E_0} \tag{3.161}$$

$$a_{12} = \frac{F_{1,2} - F_{1,0}}{\Delta E_0} \tag{3.162}$$

$$a_{13} = F_{1,0} - a_{11} \cdot E_{1,0} - a_{12} \cdot E_{2,0} \tag{3.163}$$

Ebene2:

$$F_2(E_1;E_2) = a_{21} \cdot E_1 + a_{22} \cdot E_2 + a_{23} \tag{3.164}$$

mit

$$a_{21} = \frac{F_{2,1} - F_{2,0}}{\Delta E_0} \tag{3.165}$$

$$(3.166) \qquad a_{22} = \frac{F_{2,2} - F_{2,0}}{\Delta E_0}$$

$$(3.167) \qquad a_{23} = F_{2,0} - a_{21} \cdot E_{1,0} - a_{22} \cdot E_{2,0}$$

Die Schnittgeraden S_1 und S_2 der tangentialen Ebenen 1 und 2 mit den entsprechenden Höhenebenen $F_1(J_T)$ und $F_2(J_T)$ sind dann gegeben durch

$$(3.168) \qquad F_1(J_T) = F_1(E_1,E_2) = a_{11} \cdot E_1 + a_{12} \cdot E_2 + a_{13}$$

$$(3.169) \qquad F_2(J_T) = F_1(E_1,E_2) = a_{21} \cdot E_1 + a_{22} \cdot E_2 + a_{23}$$

Der Kreuzungspunkt der beiden Schnittgeraden ist dann gegeben durch

$$(3.170) \qquad E_{1,3} = \frac{\left(F_1(J_T) - a_{13}\right) \cdot a_{22} - \left(F_2(J_T) - a_{23}\right) \cdot a_{12}}{a_{11} \cdot a_{22} - a_{12} \cdot a_{21}}$$

$$(3.171) \qquad E_{2,3} = \frac{\left(F_2(J_T) - a_{23}\right) \cdot a_{11} - \left(F_1(J_T) - a_{13}\right) \cdot a_{21}}{a_{11} \cdot a_{22} - a_{12} \cdot a_{21}}$$

Der Wellenverlagerungspunkt $[E_{1,3};E_{2,3}]$ stellt die verbesserte Näherungslösung gegenüber der vorhergehenden Näherungslösung $[E_{1,0};E_{2,0}]$ dar.

3.4.7.4 Berechnung der Verlagerungsbahn in kartesischen Koordinaten E_1, E_2 oder in Polarkoordinaten E, X_E ?

In den vorhergehenden Abschnitten 3.4.7.1 bis 3.4.7.3 wurde davon ausgegangen, dass die Berechnung der Verlagerungsbahn in kartesischen Koordinaten $E_1(J_T)$, $E_2(J_T)$ erfolgt und dementsprechende Formeln angegeben. Das ist aber auch in den Polarkoordinaten $E(J_T)$, $X_E(J_T)$ möglich. Der Ablauf und die erforderlichen Formeln sind prinzipiell gleich. Die unbekannten Variablen E_1 und E_2 müssen nur durch die neuen unbekannten Variablen E und X_E ersetzt werden. Deshalb werden der Ablauf und die Formeln für polare Koordinaten hier nicht angegeben.

Prinzipiell funktionieren beide Verfahren in wesentlichen Bereichen des Lagerspiels gleich gut und dort ist es egal, welches Verfahren verwendet wird. Für beide Verfahren gibt es aber auch jeweils einen Bereich, wo das eine bzw. das andere Verfahren Probleme bekommt. Bild 3.53 zeigt 2 Beispiele, wo jeweils eine der beiden Iterationen Probleme hat.

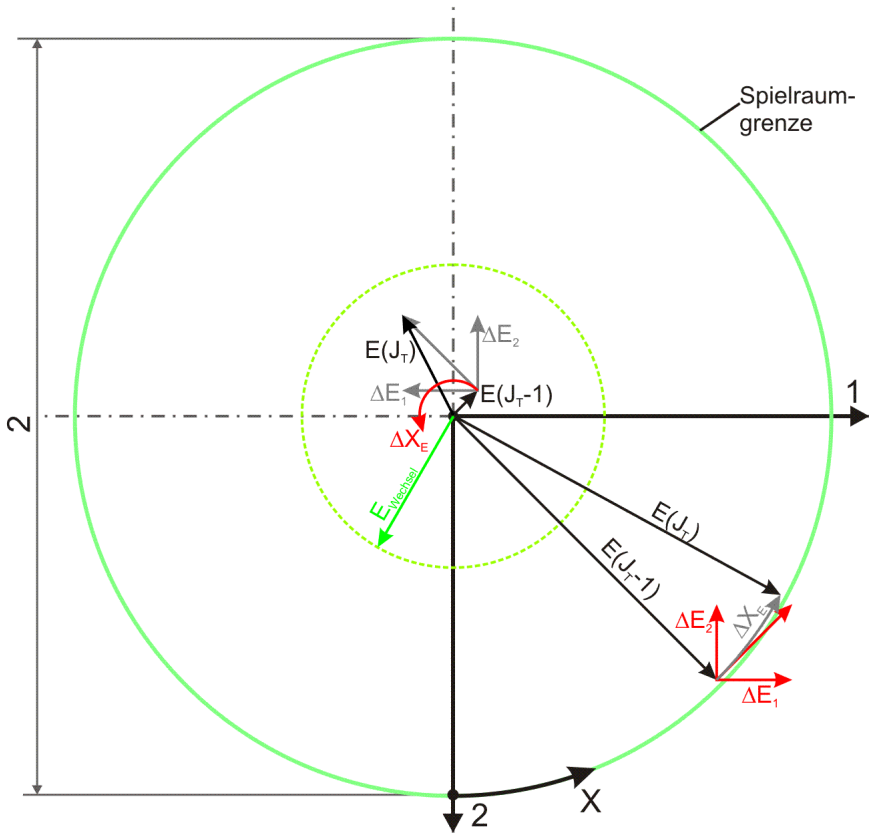


Bild 3.53: *Zwei Beispiele mit problematischer Iteration in kartesischen bzw. polaren Koordinaten*

Bei großer Exzentrizität und umlaufender Lagerbelastung, wo sich große ΔX_E und sehr kleine ΔE ergeben, können mit polaren Koordinaten noch gute Ergebnisse erzielt werden, während mit kartesischen Koordinaten die Iteration nur in sehr kleinen Schritten vorankommt, da die Tendenz besteht, dass die Iteration den Definitionsbereich über die Spielraumgrenze verlässt.

Bei sehr kleiner Exzentrizität, nahe dem singulären Zentrum des Polarkoordinatensystems, kann die Iteration mit Polarkoordinaten leicht aus dem Tritt geraten und Unsinn rechnen, was auch zur Instabilität der Iteration und damit zum Abbruch der Berechnungen führen kann.

Deshalb werden in SIRIUS wahlweise das eine bzw. das andere Verfahren angewendet. Der Anwender braucht sich darum nicht zu kümmern. Die Auswahl erfolgt durch das Programm automatisch. Mit dem programminternen Parameter $E_{Wechsel}$ (siehe Bild 3.53) ist eine Grenze festgelegt. Innerhalb dieses Grenzkreises arbeitet SIRIUS mit kartesischen Koordinaten und außerhalb mit Polarkoordinaten. Der Parameter $E_{Wechsel}$ kann vorübergehend im Menü 4.4.12.3 "Programminterne Parameter bearbeiten" verändert werden. Die Iteration mit kartesischen Koordinaten ist in der Routine "Verlagerung1" programmiert und die Iteration mit Polarkoordinaten in der Routine "Verlagerung2". Während der Berechnung einer Verlagerungsbahn kann das Programm automatisch mehrfach zwischen diesen beiden Routinen wechseln.

3.4.7.5 Vorkehrungen gegen das Überschreiten der Spielraumgrenzen

Im Abschnitt 3.4.7.2 wurde beschrieben, wie an die Funktion So=func(E,X_E) im Basispunkt $[E_0,X_{E0}]$ die Tangentialebene angelegt wird, die als Approximation der Funktion im Bereich um E_0 und X_{E0} dient und wie durch einen Schritt in Richtung ΔE und ΔX_E ein verbesserter Näherungswert für $E(J_T)$ und $X_E(J_T)$ gefunden wird. Bild 3.54 zeigt eine Skizze der Funktion So=func(E) quer durch den gesamten Spielraum des Lagers und die Tangente an die Funktion So=func(E) im Punkt E_0.

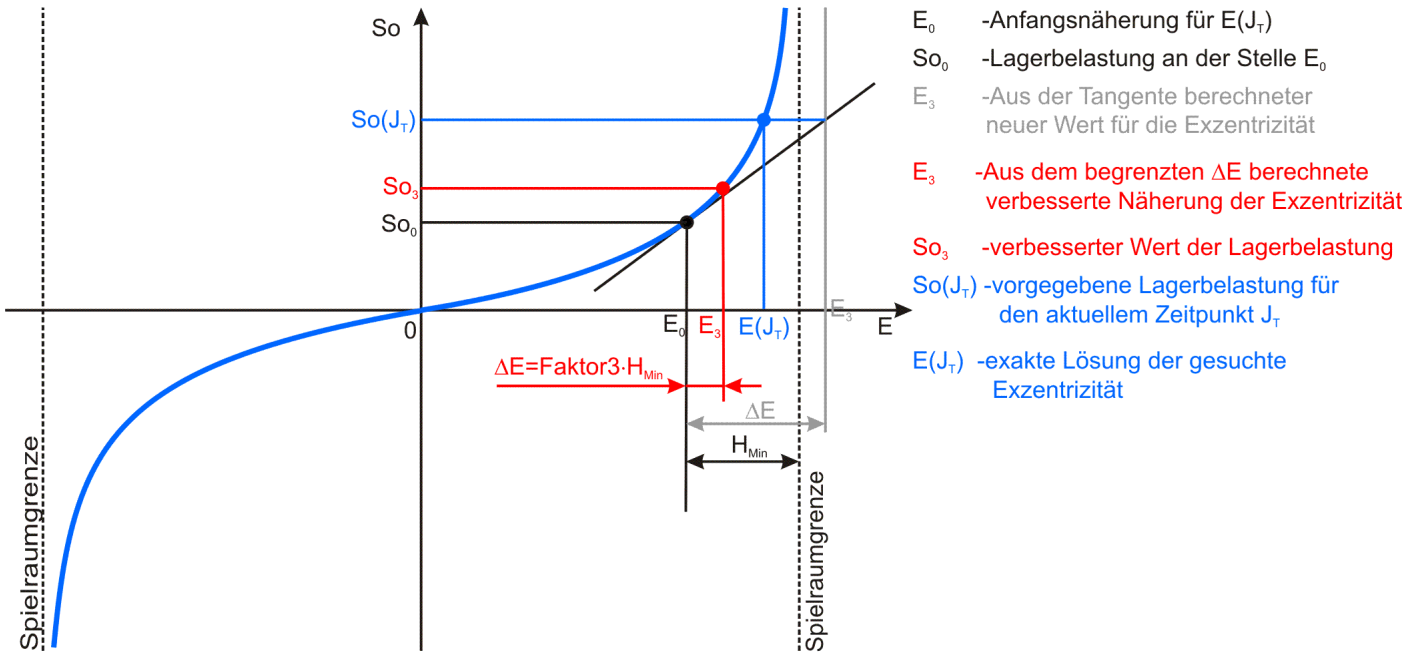


Bild 3.54: *Skizze zur Begrenzung der Schrittweite ΔE*

Da der Definitionsbereich der Funktion durch die Spielraumgrenzen begrenzt ist, kann die oben beschriebene Suchstrategie durch die Berechnung eines ΔE (grau dargestellt) aus dem Definitionsbereich herausführen. Das würde bedeuten, dass im nächsten Iterationsschritt eine Spalthöhe kleiner Null berechnet wird, was zum Abbruch der Berechnungen führt. Auch wenn ΔE so groß ist, dass die neue minimale Spalthöhe gerade noch größer Null ist, könnte sich das Ergebnis stark verschlechtern, weil mit $H_{Min} \to 0$ $So \to \infty$ geht. Deshalb wurde eine Begrenzung für ΔE eingeführt, falls sich die Exzentrizität E der Spielraumgrenze nähert.

Das Kriterium lautet:

Wenn $\Delta E > Faktor3 \cdot H_{Min}$, dann wird $\Delta E = Faktor3 \cdot H_{Min}$ gesetzt.

Ein geeigneter Faktor3 wurde empirisch ermittelt. Der Parameter **Faktor3** kann im Untermenü: 4.4.12.3 "Programminterne Parameter bearbeiten" (im Hauptmenü: "Ende der Eingabe erreicht") zeitweilig geändert werden.

3.4.7.6 Kriterien zur Beendigung der Iteration innerhalb eines Zeitschritts

Zu Berechnung der Wellenverlagerung aus einem vorgegebenen Lastverlauf werden pro Zeitpunkt J_T maximal NJ Iterationen ausgeführt. Aktuell ist die Anzahl mit NJ=10 im Programm festgelegt. Dieser Wert kann im Untermenü: "Bearbeiten der programminternen Parameter" (im Hauptmenü: "Ende der Eingabe erreicht") zeitweilig geändert werden.

Sind nach diesen NJ Iterationen die geforderten Toleranzkriterien nicht erfüllt, wird die Berechnung erfolglos mit einer Fehlermeldung abgebrochen. Siehe dazu auch Abschnitt 4.9.2.3.

Für jeden zu berechnenden Zeitpunkt ist die erste Iteration eine Extrapolation. Aus der Wellenverlagerung $E_1(J_T\text{-}1)$ und $E_2(J_T\text{-}1)$ bzw. $E(J_T\text{-}1)$ und $X_E(J_T\text{-}1)$ des vorhergehenden Zeitpunktes und deren Ableitungen $\partial E_1/\partial T$ und $\partial E_2/\partial T$ bzw. $\partial E/\partial T$ und $\partial X_E/\partial T$ werden erste Näherungswerte $E_{1,0}$ und $E_{2,0}$ bzw. E_0 und $X_{E,0}$ für die Wellenverlagerung zum Zeitpunkt J_T berechnet. Dann folgen je nach Bedarf maximal NJ-1 Iterationen, wie in den Abschnitten 3.4.7.2 bis 3.4.7.5 beschrieben.

Die ersten beiden Iterationen werden immer ausgeführt, unabhängig davon, wie genau das Ergebnis bereits ist. Beginnend nach der 2.Iteration wird anhand der Toleranzkriterien nach jeder Iteration geprüft, ob die Berechnung für den aktuellen Zeitpunkt erfolgreich beendet werden kann.

Für die Routine "Verlagerung1" wurden folgende 2 Toleranzkriterien festgelegt:

$$\sqrt{\Delta E_1^2 + \Delta E_2^2} \le H_{Min}(J_T - 1) \cdot Tol_E \qquad (3.172)$$

und

$$\sqrt{\Delta F_1^2 + \Delta F_2^2} \le Tol_F \qquad (3.173)$$

Für die Routine "Verlagerung2" wurden folgende 3 Toleranzkriterien festgelegt:

$$\left|\Delta E\right| \le H_{Min} \cdot Tol_E \qquad (3.174)$$

$$\left|\Delta X_E\right| \le Tol_{XE} \qquad (3.175)$$

und

$$\sqrt{\Delta F_1^2 + \Delta F_2^2} \le Tol_F \qquad (3.173)$$

Für eine erfolgreiche Beendigung der Berechnung müssen jeweils alle 2 bzw. 3 Kriterien erfüllt sein.

Die Parameter $\mathbf{Tol_E}$, $\mathbf{Tol_{XE}}$ und $\mathbf{Tol_F}$ können im Untermenü: 4.4.12.3 "Programminterne Parameter bearbeiten" (im Hauptmenü: "Ende der Eingabe erreicht") zeitweilig geändert werden.

Um den Verlauf der Iterationen verfolgen zu können, werden die Parameter ΔE_1, ΔE_2 und ΔF_1, ΔF_2 in der Routine "Verlagerung1" bzw. ΔE, ΔX_E, ΔF_1 und ΔF_2 in der Routine "Verlagerung2" für jeden Iterationsschritt angezeigt. Die Werte haben folgende Bedeutung:

$$\Delta E_1 = E_{1,3} - E_{1,0} \qquad (3.177)$$

$$\Delta E_2 = E_{2,3} - E_{2,0} \qquad (3.178)$$

$$\Delta E = E_3 - E_0 \qquad (3.179)$$

$$\Delta X_E = X_{E,3} - X_{E,0} \qquad (3.180)$$

$$\Delta F_1 = F_{1,3} - F_1(J_T) \qquad (3.181)$$

$$\Delta F_2 = F_{2,3} - F_2(J_T) \qquad (3.182)$$

Die Bedeutung des jeweils 2.Indizes der Parameter ist im Bild 3.52 zu erkennen. Der Index 0 steht hier für den Näherungswert des jeweiligen Parameters vor dem Iterationsschritt und der Index 3 steht für den verbesserten Näherungswert des jeweiligen Parameters nach dem Iterationsschritt. Zu beachten ist hier, das die Werte ΔF_1 und ΔF_2 hier den tatsächlich noch vorhandenen Fehler zwischen dem vorgegebenen Soll-Wert und dem in der Berechnung realisierten Wert angeben, während die Werte ΔE_1 und ΔE_2 bzw. ΔE und ΔX_E nur die letzte Verbesserung angeben. Der noch vorhandene Fehler zum exakten Wert kann also auch größer sein.

3.4.8 Eingabe der zeitlich variablen Parameter (Routine "VarPara") (unbearbeitet)

...

3.4.9 Sichern und wieder einlesen der Primärdaten (Routinen "AusgabePara5" und "LesenPara5") (unbearbeitet)

...

3.5 Ausblick auf Weiterentwicklungsmöglichkeiten (unbearbeitet)

...

Tabellenverzeichnis:

Abbildungsverzeichnis