

In Abschn. 3.3 sind mathematische Modelle für Strömungen Newtonscher Fluide vorgestellt worden. In diesem Kapitel werden nun die Lösungsverfahren diskutiert, mit denen (3.17)–(3.19) für das allgemeine Problem bzw. (3.20) und (3.21) für eine inkompressible, isotherme Strömung in CFD-Modellen üblicherweise gelöst werden.

5.1 Lösungsansätze

Bei den Lösungsverfahren wird zwischen druck- und dichtebasierten Algorithmen unterschieden. In Tab. 5.1 sind die wesentlichen Elemente dieser unterschiedlichen Lösungsansätze in einer Kurzübersicht zusammengefasst.

In druckbasierten Algorithmen werden zuerst die Strömungsgrößen \underline{u} , p , h und T berechnet. Da p nicht direkt aus den Modellgleichungen bestimmt werden kann, müssen spezielle numerische Verfahren genutzt werden. Einige dieser Verfahren werden in Abschn. 5.2 beschrieben. Ist ρ ebenfalls eine variable Strömungsgröße, wird sie über die thermische Zustandsgleichung aus p und T ermittelt. Druckbasierte Algorithmen werden

Tab. 5.1 Lösungsalgorithmen für kompressible und inkompressible Strömungen Newtonscher Fluide

druckbasierte Löser	dichtebasierte Löser
ρ konstant oder aus (3.8)	ρ aus (3.17)
\underline{u} aus (3.21) oder (3.18)	\underline{u} aus (3.18)
h , T aus Gln. (3.19), (3.9)	h , T aus Gln. (3.19), (3.9)
$p \rightarrow$ Abschn. 5.2	p aus (3.8)
sequentielle oder gekoppelte Lösung	gekoppelte Lösung

vor allem in CFD-Simulationen von inkompressiblen Strömungen eingesetzt. Sie können aber auch zur Lösung der kompressiblen Strömungsgleichungen genutzt werden.

Bei dichte-basierten Lösern werden dagegen zuerst die Strömungsgrößen ρ , \underline{u} , h und T errechnet, während hier p in einem Zwischenschritt aus ρ und T bestimmt wird. Dichte-basierte Algorithmen werden fast ausschließlich zur Berechnung kompressibler Strömungen eingesetzt. Weitere Einzelheiten können der Literatur entnommen werden, in diesem Buch werden sie jedoch nicht ausführlicher beschrieben.

Die iterative Lösung des Gleichungssystems erfolgt dann entweder gekoppelt oder sequentiell. Bei der gekoppelten Lösung werden die Gleichungen des CFD-Modells gleichzeitig gelöst, alle Unbekannten können also in einer Matrix \underline{A} entsprechend (4.70) zusammengefasst werden. Gekoppelte Lösungsverfahren werden in der Regel eingesetzt, wenn die drei Modellgleichungen der Strömung stark gekoppelt sind, wie etwa bei kompressiblen Strömungen.

Bei der sequentiellen Lösung werden die Gleichungen des mathematischen Modells dagegen nacheinander gelöst. In diesem Fall wird für jede unbekannte Strömungsgröße ϕ jeweils eine eigene Matrix \underline{A}_ϕ entsprechend (4.70) formuliert und gelöst. Durch sogenannte äußere Iterationen wird berücksichtigt, dass die einzelnen Gleichungen miteinander gekoppelt sind. Sequentielle Lösungsverfahren werden üblicherweise eingesetzt, wenn die Modellgleichungen nur schwach miteinander gekoppelt sind, beispielsweise bei inkompressiblen Strömungen.

5.2 Druckbasierte Algorithmen

5.2.1 Übersicht

Druckbasierte Algorithmen nutzen überwiegend gekoppelte Lösungsverfahren, sequentielle Lösungsverfahren mit einer Druck-Poisson-Gleichung oder sequentielle Lösungsverfahren mit einer Druckkorrektur-Gleichung. Bei der sequentiellen Lösung der Grundgleichungen ergibt sich das oben angesprochene Problem, dass die gesuchte Strömungsgröße p nicht direkt aus einer der Modellgleichungen bestimmt werden kann. Deshalb muss für diese Verfahren eine weitere Modellgleichung für p entwickelt werden. Als Modellgleichung kann dabei entweder die Druck-Poisson- oder eine Druckkorrektur-Gleichung genutzt werden.

5.2.2 Verfahren mit Druck-Poisson-Gleichung

Die Druck-Poisson-Gleichung als Modellgleichung für den Druck ergibt sich, indem die Divergenz der Navier-Stokes-Gleichung gebildet wird. Bei inkompressiblen Strömungen

folgt aus (3.21)

$$\begin{aligned} \rho \nabla \cdot \left[\frac{\partial \underline{u}}{\partial t} + \nabla \cdot (\underline{u} \underline{u}) \right] &= \nabla \cdot \left[-\nabla p + \eta \Delta \underline{u} + \rho \underline{g} \right] \\ \Delta p &= \nabla \cdot \left[\eta \Delta \underline{u} - \rho \nabla \cdot (\underline{u} \underline{u}) - \rho \frac{\partial \underline{u}}{\partial t} \right] \end{aligned} \quad (5.1)$$

Für die konvergierte Lösung einer inkompressiblen Strömung vereinfacht sich diese Gleichung zu

$$\Delta p = -\rho \nabla \cdot [\nabla \cdot (\underline{u} \underline{u})] \quad (5.2)$$

da dann auch die Kontinuitätsgleichung (3.20) erfüllt sein muss. In sequentiellen Lösungsverfahren müssen aber Druckgleichungen in der Form von (5.1) gelöst werden, da während der iterativen Annäherung an die gesuchte Lösung die Kontinuitätsgleichung (3.20) noch nicht erfüllt sein muss. Da die korrekte Implementierung von (5.1) außerdem mit größerem numerischen Aufwand verbunden ist, werden in vielen CFD-Programmen stattdessen die im nächsten Abschnitt diskutierten Druckkorrektur-Verfahren genutzt.

5.2.3 Druckkorrektur-Verfahren

Druckkorrektur-Gleichung In einem Druckkorrektur-Verfahren sind die diskretisierte Kontinuitäts- und Navier-Stokes-Gleichung, beispielsweise für eine stationäre, inkompressible, isotherme Strömung

$$[\nabla \cdot \underline{u}]_{KV} = 0 \quad (5.3)$$

$$\rho [\nabla \cdot (\underline{u} \underline{u})]_{KV} = -[\nabla p]_{KV} - \eta [\Delta \underline{u}]_{KV} \quad (5.4)$$

in jedem äußeren Iterationsschritt (n), ($n + 1$), usw. erfüllt. In den Gleichungen steht $[\dots]_{KV}$ für die FVM-Diskretisierung der jeweiligen Terme in einem KV. Die prinzipielle Schwierigkeit bei der sequentiellen Lösung von (5.3) und (5.4) besteht nun darin, dass $\underline{u}^{(n+1)}$ und $p^{(n+1)}$ nicht gleichzeitig bekannt sind, sondern nacheinander bestimmt werden müssen. In einem Druckkorrektur-Verfahren wird deshalb ein innerer Prädiktor-Korrektor-Algorithmus genutzt, um die beiden gesuchten Größen nacheinander zu berechnen.

Wenn $\underline{u}^{(n)}$ und $p^{(n)}$ im Iterationsschritt (n) bekannt sind, werden zuerst die nächsten Werte $\underline{u}^{(*)}$ und $p^{(*)}$ geschätzt, danach werden Korrekturen $\underline{u}^{(K)}$ und $p^{(K)}$ bestimmt. Schließlich ergeben sich die Werte $\underline{u}^{(n+1)}$ und $p^{(n+1)}$ im nächsten Iterationsschritt ($n + 1$) aus

$$\underline{u}^{(n+1)} = \underline{u}^{(*)} + \underline{u}^{(K)} \quad (5.5)$$

$$p^{(n+1)} = p^{(*)} + p^{(K)} \quad (5.6)$$

Ein einfacher Prädiktor-Korrektor-Algorithmus umfasst folgende Punkte

1. Eine Druckschätzung $p^{(*)}$, etwa durch

$$p^{(*)} = p^{(n)} \quad (5.7)$$

2. Die Schätzung der Geschwindigkeit $\underline{u}^{(*)}$, beispielsweise mit (5.4) in der Formulierung

$$\rho \left[\nabla \cdot \left(\underline{u}^{(*)} \underline{u}^{(*)} \right) \right]_{\text{KV}} = - \left[\nabla p^{(n)} \right]_{\text{KV}} - \eta \left[\Delta \underline{u}^{(*)} \right]_{\text{KV}} \quad (5.8)$$

3. Die Formulierung des Zusammenhangs zwischen Geschwindigkeits- und Druckkorrektur, die ebenfalls aus (5.4) abgeleitet werden kann

$$\rho \left[\nabla \cdot \left(\underline{u}^{(K)} \underline{u}^{(K)} \right) \right]_{\text{KV}} = - \left[\nabla p^{(K)} \right]_{\text{KV}} - \eta \left[\Delta \underline{u}^{(K)} \right]_{\text{KV}}$$

Die Auswertung der nach der Diskretisierung vorliegenden algebraischen Gleichung ermöglicht es, die Geschwindigkeits- durch die Druckkorrektur auszudrücken

$$\underline{u}^{(K)} = \underline{f} \left(p^{(K)} \right) \quad (5.9)$$

4. Die Formulierung der Druckkorrektur-Gleichung, die aus $\underline{u}^{(*)}$, $\underline{u}^{(K)}$ und der Kontinuitätsgleichung

$$\left[\nabla \cdot \underline{u}^{(n+1)} \right]_{\text{KV}} = \left[\nabla \cdot \underline{u}^{(*)} \right]_{\text{KV}} + \left[\nabla \cdot \underline{u}^{(K)} \right]_{\text{KV}} = 0$$

folgt. Wird $\underline{u}^{(K)}$ durch $p^{(K)}$ ausgedrückt

$$0 = \left[\nabla \cdot \underline{u}^{(*)} \right]_{\text{KV}} + \left[\nabla \cdot \underline{f} \left(p^{(K)} \right) \right]_{\text{KV}}$$

kann die Druckkorrektur-Gleichung bestimmt werden

$$p^{(K)} = h \left(\underline{u}^{(*)} \right) \quad (5.10)$$

Sie beschreibt, wie $p^{(K)}$ in den einzelnen Stützstellen des Gitters aus den diskreten Werten von $\underline{u}^{(*)}$ in diesen Stützstellen errechnet werden kann.

Ablauf Die Gleichungen (5.5)–(5.10) werden sequentiell in folgenden Schritten gelöst

1. Startwerte $\underline{u}^{(n)}$ und $p^{(n)}$ sind bekannt, Druck mit (5.7) schätzen
2. Geschwindigkeit $\underline{u}^{(*)}$ im Prädiktor-Schritt mit (5.8) bestimmen
3. Druckkorrektur $p^{(K)}$ gemäß (5.10) berechnen
4. Geschwindigkeitskorrektur $\underline{u}^{(K)}$ gemäß (5.9) bestimmen
5. Endwerte $\underline{u}^{(n+1)}$ und $p^{(n+1)}$ entsprechend (5.5) und (5.6) ermitteln

Einzelne Verfahren Das in der CFD bisher am häufigsten genutzte Druckkorrektur-Verfahren ist der SIMPLE-Algorithmus [49] (SIMPLE für Semi Implizit Method for Pressure Linked Equations). SIMPLE nutzt eine Druckkorrektur-Gleichung (5.10), in der einige unbekannte Terme, die sich aus der Diskretisierung und dem Ablauf des Algorithmus ergeben, vernachlässigt werden. Aufgrund dieser Vereinfachungen ist eine Unterrelaxation sowohl der Geschwindigkeit als auch des Drucks notwendig, um die Konvergenz des iterativen Lösungsprozesses zu gewährleisten. Die Strömungsgrößen zum Iterationsschritt $(n + 1)$ werden daher wie folgt berechnet

$$\underline{u}^{(n+1)} = \alpha_u \left(\underline{u}^{(*)} + \underline{u}^{(K)} \right) + (1 - \alpha_u) \underline{u}^{(n)} \quad (5.11)$$

$$p^{(n+1)} = p^{(n)} + \alpha_p p^{(K)} \quad (5.12)$$

Darin sind α_u und α_p die Unterrelaxationsfaktoren für die Geschwindigkeit und den Druck.

SIMPLE-Algorithmus

Die Vorstellung des SIMPLE-Algorithmus durch S. H. Patankar [48] war sicher einer der wichtigsten Beiträge in der Entwicklung von CFD. Noch heute ist Patankar einer der am häufigsten zitierten Autoren in Naturwissenschaft und Technik (aktuell wird in etwa 17500 Arbeiten auf das Buch [48] verwiesen).

In der Literatur sind verschiedene Erweiterungen des SIMPLE-Algorithmus vorgestellt worden, um die Leistungsfähigkeit des Druckkorrektur-Verfahrens für bestimmte Strömungskonfigurationen zu verbessern. Eine solche Variante ist der SIMPLEC-Algorithmus [10] (SIMPLEC für SIMPLE Corrected). In diesem Verfahren werden die in SIMPLE vernachlässigten Terme beibehalten und ihre Beiträge durch bekannte Werte approximiert. Der SIMPLEC-Algorithmus benötigt deswegen keine Unterrelaxation.

Ein weiteres, in CFD-Simulationen häufig genutztes Druckkorrektur-Verfahren ist der PISO-Algorithmus [28] (PISO für Pressure-Implicit Split-Operator). Die PISO-Schleife beginnt wie der SIMPLE-Algorithmus mit einem Prädiktor- und einem ersten Korrektorschritt, wobei in diesem ersten Schritt die gleichen Druck- und Geschwindigkeitskorrekturen wie in SIMPLE genutzt werden. Danach wird im PISO-Algorithmus aber ein zweiter Korrektorschritt durchlaufen, aus dem sich zweite Korrekturen $\underline{u}^{(2K)}$, $p^{(2K)}$ ergeben. Außerdem werden die im ersten Schritt vernachlässigten Terme der Druckkorrektur-Gleichung im zweiten Korrektorschritt auf Basis der Geschwindigkeitskorrektur $\underline{u}^{(K)}$ approximiert und berücksichtigt. Schließlich werden die Endwerte $\underline{u}^{(n+1)}$ und $p^{(n+1)}$ unter Berücksichtigung von $\underline{u}^{(K)}$ und $\underline{u}^{(2K)}$ bzw. $p^{(K)}$ und $p^{(2K)}$ berechnet.

5.3 Praktikum: Nischenströmung

Mit Hilfe der in Abschn. 5.2 vorgestellten Druckkorrektur-Verfahren soll jetzt die zweidimensionale Nischenströmung in CFD-Simulationen berechnet werden.

Problembeschreibung In Abb. 5.1 ist die Konfiguration dieser Strömung skizziert. An drei feststehenden Wänden (links, rechts und unten) gilt die Haftbedingung $\underline{u} = 0$, der Deckel wird dagegen mit endlicher Geschwindigkeit $u_x = u_D$ bewegt. In der Nische stellt sich dadurch eine rezirkulierende Strömung ein, in den Ecken bilden sich außerdem sekundäre Rezirkulationsgebiete aus.

Für dieses Strömungsproblem gibt es eine Vielzahl von Referenzlösungen in der Literatur, siehe zum Beispiel die Arbeit von Ghia et al. [18]. Später werden diese Daten genutzt, um die hier erzielten Ergebnisse zu überprüfen.

Die Modellgleichungen der zweidimensionalen stationären laminaren Strömung lauten

$$\nabla \cdot \underline{u} = 0 \quad (5.13)$$

$$\rho [\nabla \cdot (\underline{u} \underline{u})] = -\nabla p + \eta (\Delta \underline{u}) \quad (5.14)$$

5.3.1 Lösung mit OpenFOAM

Führen Sie die CFD-Simulation in OpenFOAM in folgenden Schritten durch, verwenden Sie dabei den für diese Problemstellung geeigneten Löser `simpleFoam`. Beachten Sie auch die Hinweise zum Praktikum *Konvektion eines Skalars* in Abschn. 4.3.

Abb. 5.1 Konfiguration der zweidimensionalen Nischenströmung

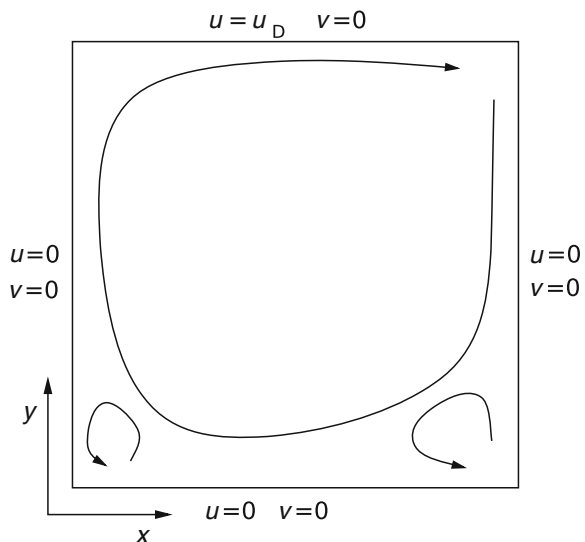


Abb. 5.2 Nischenströmung
mit OpenFOAM: Laminare
Strömung

```

a
RASModel      laminar;
turbulence     off;
printCoeffs    on;
Eintragungen in RASProperties

b
transportModel Newtonian;
nu              nu [ 0 2 -1 0 0 0 0 ] 0.0001;
Eintragungen in transportProperties

```

Lösungshinweise

1. Bereiten Sie die Simulation vor, indem Sie die Verzeichnisse `constant/`, `system/` und `0/` anlegen. Durch Eintragungen im Verzeichnis `constant/polyMesh` wird das Gitter mit 200×200 Zellen wie Praktikum *Ebenes, quadratisches Strömungsgebiet*, Abschn. 2.5, generiert.
2. Konzipieren Sie das numerische Modell: Spezifizieren Sie das stationäre Problem durch eine Wahl von u_D , ρ und η so, dass die Reynolds-Zahl $Re = \rho u_D L / \eta = 1000$ erreicht wird, wenn L die Kantenlänge des quadratischen Strömungsgebiets ist. Passende Angaben sind zum Beispiel $u_D = 1 \text{ m/s}$ und $\nu = 10^{-4} \text{ m}^2/\text{s}$. Da nur Re betrachtet wird, müssen Ihre Stoffwerte zu keinem konkreten Fluid passen.
3. Im Verzeichnis `constant/` werden die Dateien `RASProperties` und `transportProperties` vorläufig wie im Tutorial `airFoil2D` für den Löser `simpleFoam` (zu finden in `../OpenFOAM-2.0.x/tutorials/incompressible/`) angelegt.
4. Spezifizieren Sie die laminare Strömung durch den Eintrag `laminar` bei `RASmodel` in der Datei `RASProperties`, Abb. 5.2a.
5. Geben Sie das Newtonsche Fließgesetz (3.11) und die kinematische Viskosität des Newtonschen Fluids durch entsprechende Einträge bei `transportModel` und `nu` in der Datei `transportProperties` an, Abb. 5.2b.
6. Im Verzeichnis `system/` werden die Dateien `controlDict`, `fvSchemes` und `fvSolution` ebenfalls vorläufig wie in einem der Tutorials für den Löser `simpleFoam` angelegt.
7. Wählen Sie die numerischen Verfahren für das Problem durch entsprechende Eintragungen in `fvSchemes`. Spezifizieren Sie die stationäre Strömung durch den Eintrag `steadyState`. Wählen Sie die LUDS-Interpolation (`linearUpwindV`) zur Diskretisierung des konvektiven Terms, die anderen Eintragungen in `fvSchemes` müssen nicht geändert werden, Abb. 5.3a.
8. Öffnen Sie die Datei `fvSolution`. Die Eintragungen zeigen Folgendes an: Zur Lösung des Systems der Druck-Gleichungen entsprechend (4.70) wird ein Multigrid-Verfahren, siehe Abschn. 4.2, genutzt, das über den Eintrag `GAMG` bei `p` aufgerufen wird, Abb. 5.4a. Übernehmen Sie diese Eintragungen auch für die

Abb. 5.3 Nischenströmung mit OpenFOAM: Diskretisierungsverfahren und Iterationsparameter

```

a
ddtSchemes
{
    default      steadyState;
}

gradSchemes
{
    default      Gauss linear;
}

divSchemes
{
    default      none;
    div(phi,U)   Gauss linearUpwindV grad(U);
    div((nuEff*dev(T(grad(U)))) Gauss linear;
}

Eintragungen in fvSchemes

b
application      simpleFoam;
startFrom        startTime;
startTime        0;
stopAt           endTime;
endTime          2000;
deltaT           1;

Eintragungen in controlDict

```

Lösung der Gleichungssysteme für die Geschwindigkeitskomponenten u bzw. v (Aufruf bei U).

9. Die u -, v - und p -Gleichungen sind entsprechend dem SIMPLE-Algorithmus, siehe Abschn. 5.2.3, gekoppelt (Eintragung SIMPLE), die Unterrelaxation nach (5.11) und (5.12) erfolgt mit den Parametern $\alpha_u = 0,7$ und $\alpha_p = 0,3$ (Eintragungen bei `relaxationsFactors`). Diese Werte können beibehalten werden. Setzen Sie die Abbruchkriterien R_{Ab}^ϕ für die Residuen für p , u und v jeweils auf 10^{-5} , geben Sie außerdem eine Referenzzelle und einen Referenzwert für p an, Abb. 5.4c (Eintragungen bei `residualControl`).
10. Setzen Sie in `controlDict` den Startpunkt der Iteration auf 0 (Eintragung bei `startFrom` und `startTime`) und die maximale Zahl an Iterationsschritten auf 5000 (Eintragung bei `stopAt` und `endTime`), Abb. 5.3b.
11. Geben Sie die Randbedingungen für das Geschwindigkeitsfeld entsprechend Abb. 5.1 vor, indem Sie die Datei `0/U` anlegen und dort auf den jeweiligen Rändern die passenden Angaben machen, Abb. 5.5a. Legen Sie außerdem die Datei `0/p` an, dort geben Sie auf allen Rändern die Bedingung $\nabla_n p = 0$ vor, Abb. 5.5b.
12. Führen Sie die CFD-Simulation aus, indem Sie `simpleFoam` starten. Notieren Sie nach Beendigung der Simulation die benötigten Iterationsschritte und die benötigte Rechenzeit (Execution Time).
13. Untersuchen Sie folgende Fragen:
 - Wie wirkt sich ein Wechsel bei den Lösungsverfahren für die Gleichungssysteme vom Multigrid- zu Gradienten-Verfahren auf die benötigten Iterationsschritte und die benötigte Rechenzeit aus? Die Gradienten-Verfahren werden wie in

Abb. 5.4 Nischenströmung
mit OpenFOAM: Eintragungen
in fvSolution

```
a
solvers
{
    p
    {
        solver          GAMG;
        tolerance       1e-06;
        relTol          0.1;
        smoother        GaussSeidel;
        nPreSweeps       0;
        nPostSweeps      2;
        cacheAgglomeration true;
        nCellsInCoarsestLevel 10;
        agglomerator     faceAreaPair;
        mergeLevels      1;
    }
}
```

Spezifizierung Multigrid-Verfahren

```
b
p
{
    solver          PCG;
    preconditioner   DIC;
    tolerance       1e-06;
    relTol          0;
}

U
{
    solver          PBiCG;
    preconditioner   DILU;
    tolerance       1e-05;
    relTol          0;
}
```

Spezifizierung Gradientenverfahren

```
c
SIMPLE
{
    nNonOrthogonalCorrectors 0;

    pRefCell      0;
    pRefValue     0;

    residualControl
    {
        p          1e-5;
        U          1e-5;
    }
}

relaxationFactors
{
    default       0;
    p             0.3;
    U             0.7;
}
```

Druckkorrektur-Verfahren

Abb. 5.4b dargestellt durch die Eintragung PCG bei p bzw. PBiCG bei U in der Datei fvSolution aufgerufen.

- Wie viele Iterationsschritte und welche Rechenzeit ist erforderlich, wenn Sie die Simulationen mit dem Multigrid- und den Gradienten-Verfahren auch auf den beiden anderen im Praktikum *Ebenes, quadratisches Strömungsgebiet* generierten Gittern ausführen?
- Wie sieht das Ergebnis der Strömungssimulation aus, wie gut stimmen Ihre Ergebnisse mit bekannten Daten überein?

Diskussion Der Verlauf der Residuen R_u und R_p der u - und der p -Gleichungen in den Simulationen mit dem Multigrid- (GAMG) und den Gradienten-Verfahren (CG) ist in

Abb. 5.5 Nischenströmung mit OpenFOAM: Randbedingungen

a

```

boundaryField
{
    unten
    {
        type          value
        fixedValue     uniform (0 0 0);
    }
    ...
    oben
    {
        type          value
        fixedValue     uniform (1.0 0 0);
    }
}

```

Eintragungen in U

b

```

boundaryField
{
    unten
    {
        type          zeroGradient;
    }
    ...
}

```

Eintragungen in p

Abb. 5.6 Nischenströmung mit OpenFOAM: Verlauf der Residuen, GAMG; Multigrid-Verfahren, CG: Gradienten-Verfahren

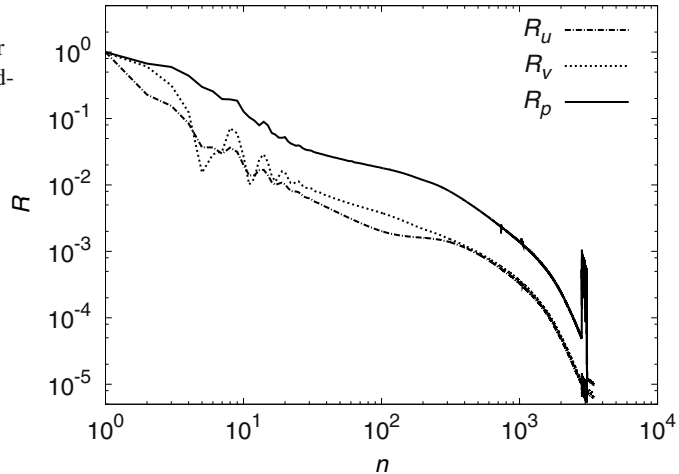


Abb. 5.6 dargestellt¹. Es ergibt sich der typische Verlauf für eine stabile iterative Lösung des Gleichungssystems. Die Residuen nehmen im Verlauf der Iteration ab und unterschreiten schließlich das vorgegebene Abbruchkriterium. Die Iteration wird dann abgebrochen und das Ergebnis der Simulation steht fest. Die Verläufe der Residuen bei den beiden unterschiedlichen Lösungsverfahren unterscheiden sich nur sehr gering.

¹ Hinweis: Falls Sie ähnliche grafische Darstellungen der Residuenverläufe generieren wollen, können Sie das OpenFOAM-Utility `foamJob` nutzen, um `simpleFoam` im Hintergrund zu starten. Die Ausgaben während der CFD-Simulation werden dann in der Datei `log` im case-Verzeichnis gespeichert. Die Datei kann anschließend mit dem OpenFOAM-Utility `foamLog` ausgewertet werden.

Tab. 5.2 Nischenströmung mit OpenFOAM: Iterationsschritte n_{Iter} und Rechenzeiten t_{comp}

Gitter	GAMG		CG	
	n_{Iter}	t_{comp}	n_{Iter}	t_{comp}
50×50 Zellen	803	4 s	772	5 s
100×100 Zellen	1789	34 s	1606	72 s
200×200 Zellen	3827	299 s	3448	1366 s

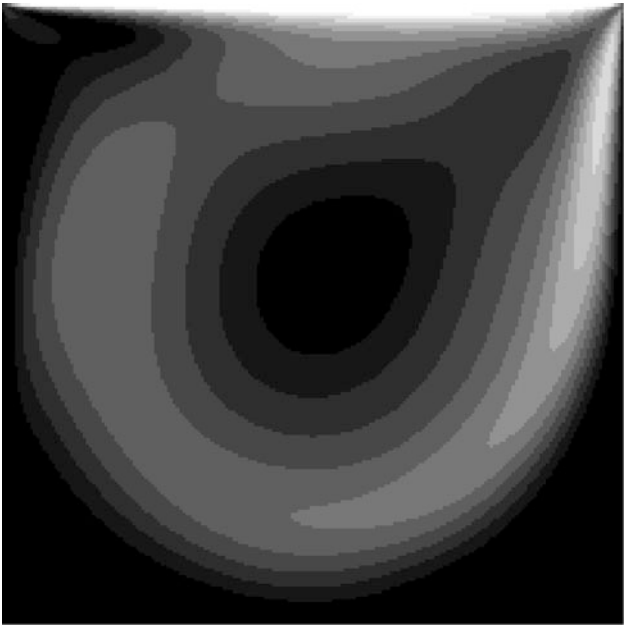
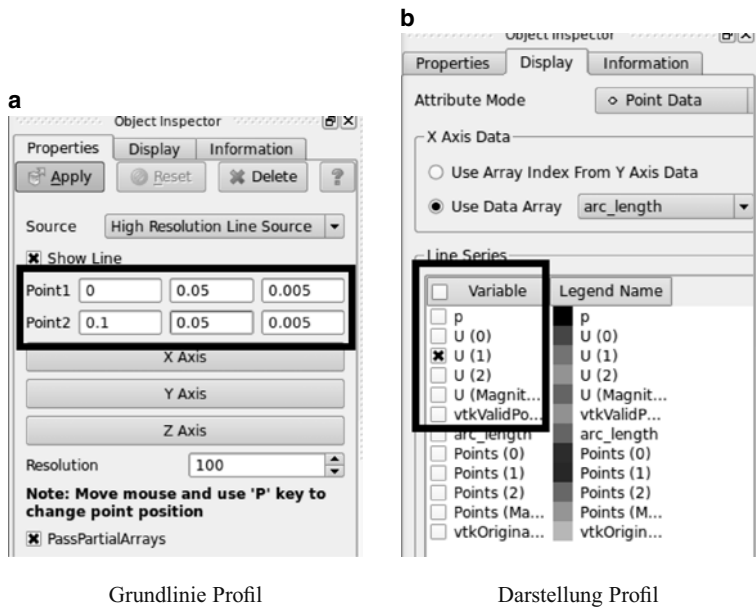


Abb. 5.7 Nischenströmung mit OpenFOAM: Geschwindigkeitsfeld

Allerdings zeigen sich merkliche Unterschiede bei der Zahl der bis zum Iterationsabbruch notwendigen Iterationsschritte n_{Iter} und bei der benötigten Rechenzeit t_{comp} , Tab. 5.2. Das Multigrid-Verfahren benötigt auf jedem Gitter etwas mehr Iterationsschritte, dafür ist die bis zum Iterationsabbruch erforderliche Rechenzeit speziell auf dem mittleren und dem feinen Gitter wesentlich geringer als bei den Simulationen mit den Gradienten-Verfahren. Dieser Befund bestätigt den generellen Trend, dass in modernen CFD-Programmen die Multigrid-Verfahren die üblichen Lösungsverfahren sind [69]. Allerdings kann es bei bestimmten Problemen durchaus von Vorteil sein, auch andere Lösungsverfahren zu erproben, um die Konvergenz des Verfahrens zu beschleunigen und so Rechenzeiten zu verkürzen.

Das in der Simulation auf dem Gitter mit 200×200 Zellen ermittelte Geschwindigkeitsfeld besitzt die erwartete Struktur, Abb. 5.7. Es existiert ein großes Rezirkulationsge-



Grundlinie Profil

Darstellung Profil

Abb. 5.8 Nischenströmung mit OpenFOAM: Auswertung Simulationsergebnisse

biet in der Nische, dabei werden die höchsten Geschwindigkeiten direkt an der bewegten oberen Wand gefunden.

Um die Ergebnisse quantitativ mit bekannten Daten, wie etwa von Ghia et al. [18], vergleichen zu können, wird im Menü *Filters > Alphabetical > Plot Over Line*, Reiter *Properties* zuerst die Grundlinie eines geeigneten Profils, in diesem Fall mit dem Startpunkt bei $x = 0\text{ m}$, $y = 0,05\text{ m}$, $z = 0,005\text{ m}$ und dem Endpunkt bei $x = 0,1\text{ m}$, $y = 0,05\text{ m}$, $z = 0,005\text{ m}$, definiert, Abb. 5.8a. Anschließend kann der Verlauf einer Geschwindigkeitskomponente entlang dieser Profillinie dargestellt werden, indem im Reiter *Display* die entsprechende Größe, hier $U(1)$ entsprechend u_y , für die Darstellung ausgewählt wird, Abb. 5.8b. In Abb. 5.9 sind das berechnete Geschwindigkeitsprofil $u_y(x, y = 0,05\text{ mm})$ (OpenFOAM) und die entsprechenden Referenzwerte aus Ghia et al. [18] dargestellt. Die Übereinstimmung zwischen den Ergebnissen aus OpenFOAM und den Referenzwerten ist offenkundig sehr gut. Vergleichen Sie auch die Profile aus den Simulationen auf den anderen Gittern mit den Referenzwerten.

5.3.2 Lösung mit ANSYS FLUENT

Führen Sie die CFD-Simulation in ANSYS FLUENT durch, gehen Sie dabei in den unten beschriebenen Schritten vor, um die Lösung zu berechnen. Beachten Sie auch die Hinweise zum Praktikum *Konvektion eines Skalars* in Abschn. 4.3.

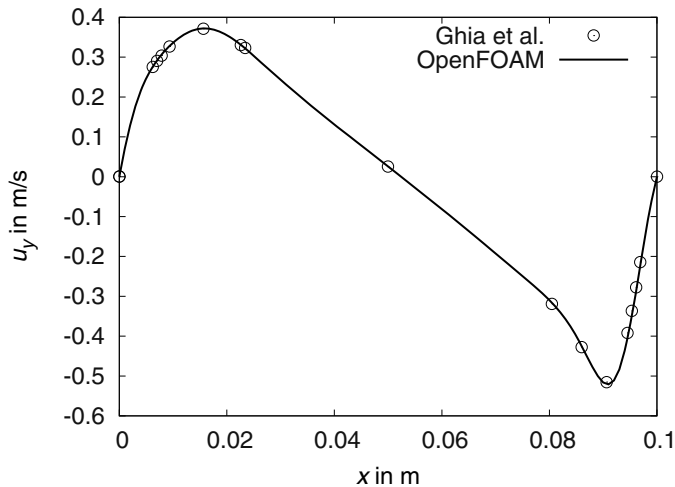
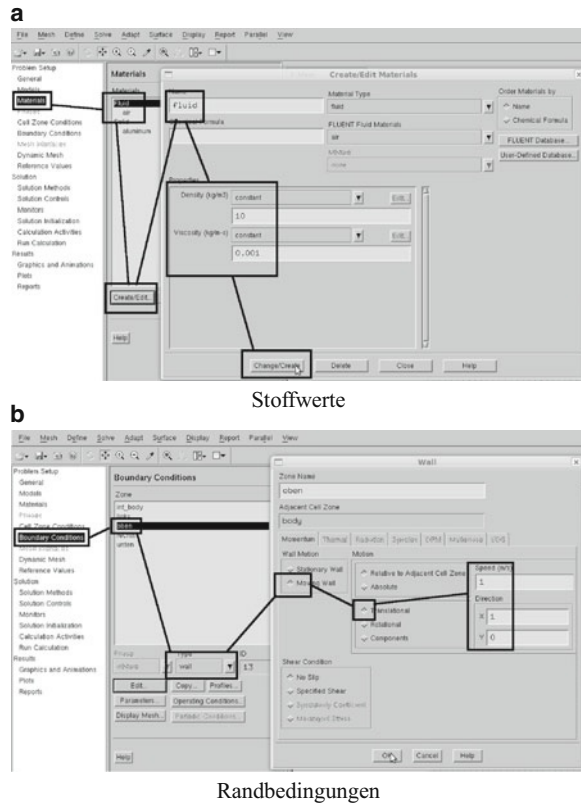


Abb. 5.9 Nischenströmung mit OpenFOAM: Geschwindigkeitsprofil $u_y(x, y = 0,05 \text{ mm})$

Lösungshinweise

1. Starten Sie ANSYS FLUENT und berücksichtigen Sie, dass Ihr Problem in 2D zu lösen ist.
2. Verwenden Sie zuerst das im Praktikum *Ebenes, quadratisches Strömungsgebiet*, Abschn. 2.5, erzeugte Gitter mit 200×200 Zellen, laden Sie dazu die Datei `QuadFlowDom-200.msh` in ANSYS FLUENT (Menü `File > Read > Mesh`). Skalieren Sie das Gitter auf die Größe $0,1 \text{ m} \times 0,1 \text{ m}$.
3. Formulieren Sie das numerische Modell: Spezifizieren Sie das stationäre Problem durch eine Wahl von u_D , ρ und η so, dass die Reynolds-Zahl $Re = \rho u_D L / \eta = 1000$ erreicht wird, wenn L die Kantenlänge des quadratischen Strömungsgebiets ist. Passende Angaben sind zum Beispiel $u_D = 1 \text{ m/s}$, $\rho = 10 \text{ kg/m}^3$ und $\eta = 10^{-3} \text{ Pa} \cdot \text{s}$, Abb. 5.10a. Da nur Re betrachtet wird, müssen Ihre Stoffwerte zu keinem konkreten Fluid passen. Setzen Sie die Randbedingungen (Zugang über `Define > Boundary Conditions ...`) entsprechend Abb. 5.1. Um eine Geschwindigkeit am Rand oben vorzugeben, editieren Sie den Rand im Menü `Problem Setup > Boundary Conditions > Zone` (der Typ `wall` muss nicht geändert werden). Wählen Sie die Optionen `Moving Wall` und `Translational` und geben Sie anschließend die Wandgeschwindigkeit mit `Speed 1` vor. Achten Sie darauf, dass die Richtung der Wandbewegung korrekt gewählt wird, sie muss entlang der x -Achse sein, Abb. 5.10b.
4. Koppeln Sie die einzelnen Gleichungen der gesuchten Strömungsgrößen p , u und v mit dem SIMPLE-Algorithmus, siehe Abschn. 5.2.3. Sie können SIMPLE über das Menü `Solution > Solution Methods > Pressure-Velocity Coupling > Scheme` anwählen. Geben Sie außerdem die LUDS-

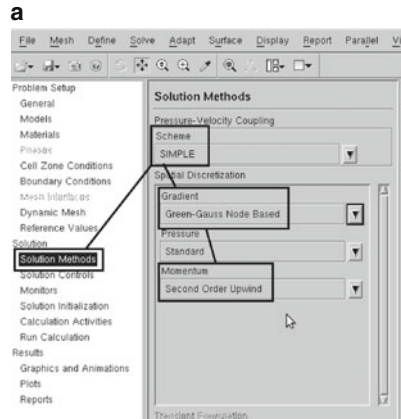
Abb. 5.10 Nischenströmung mit ANSYS FLUENT: Konfiguration



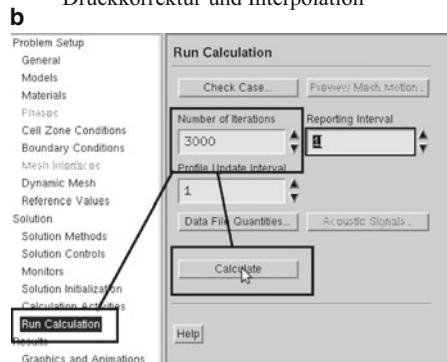
Interpolation (Second Order Upwind) bei Spatial Discretization > Momentum an, Abb. 5.11a. Zur Bestimmung der Gradienten bei der LUDS-Interpolation wird das Verfahren Green-Gauss Node Based (bei Spatial Discretization > Gradient) verwendet, das auf der numerischen Auswertung des Gaußschen Satzes beruht.

- Öffnen Sie das Menü Solution > Solution Controls. Hier sind die Parameter für die Unterrelaxation nach (5.11) und (5.12) angegeben. Die gesetzten Werte $\alpha_p = 0,3$ (Pressure) und $\alpha_u = 0,7$ (Momentum) müssen nicht geändert werden. ANSYS FLUENT nutzt Multigrid-Verfahren, siehe Abschn. 4.2, um die Gleichungssysteme für p , u und v entsprechend (4.70) zu lösen. Sie können die Einstellungen für die Multigrid-Verfahren überprüfen, indem Sie das Menü Solution > Solution Controls > Advanced ... öffnen.
- Setzen Sie die Abbruchkriterien der Residuen für u und v jeweils auf $R_{Ab}^u = R_{Ab}^v = 10^{-5}$. Um diese Werte einzutragen, öffnen Sie das Menü Monitors > Residuals - Print, Plot über den Schalter Edit ..., Die Angaben sind dann in der Rubrik Equations zu machen. Für den Druck p kann

Abb. 5.11 Nischenströmung
mit ANSYS FLUENT: Nume-
rische Parameter



Druckkorrektur und Interpolation

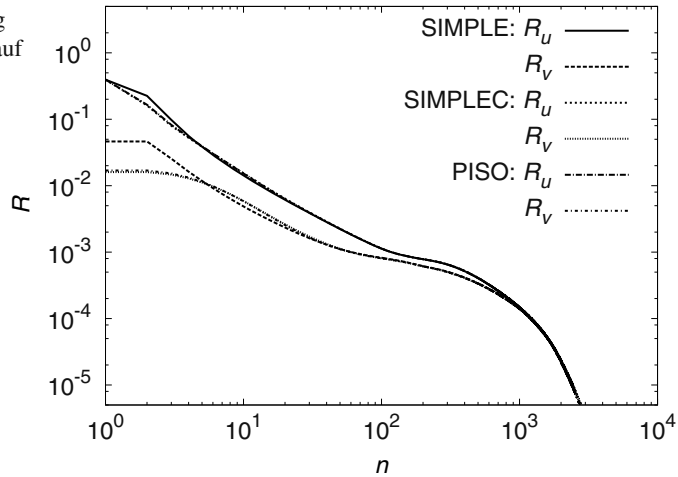


Iteration

kein Abbruchkriterium gesetzt werden, stattdessen überprüft ANSYS FLUENT das Residuum der Kontinuitätsgleichung. Der dort standardmäßig gesetzte Wert von $R_{Ab}^m = 10^{-3}$ kann beibehalten werden.

7. Initialisieren Sie p , u und v jeweils auf den Wert 0 (Menü Solution > Solution Initialization). Berechnen Sie dann die Lösung (Menü Solution > Run Calculation). Geben Sie eine hinreichend große Zahl an Iterationsschritten (Number of Iterations) vor, beispielsweise 3000, und starten Sie die Rechnung (Calculate), Abb. 5.11b. Wenn Sie eine konvergierte Lösung erreicht haben, erscheint in der Konsole der Hinweis *solution is converged*.
8. Notieren Sie nach Beendigung der Simulation die benötigten Iterationsschritte. Ermitteln Sie außerdem die benötigte Rechenzeit, indem Sie über das Textinterface (Konsolen-Eingabe) den Befehl `report > system > proc-stats` eingeben. In der Konsole erscheint dann eine Tabelle mit einigen Informationen über die Simulation, unter anderem ist die benötigte CPU-Zeit aufgelistet.

Abb. 5.12 Nischenströmung mit ANSYS FLUENT: Verlauf der Residuen



9. Untersuchen Sie folgende Fragen:

- Wie wirkt sich ein Wechsel des Druckkorrektur-Verfahrens von SIMPLE auf SIMPLEC bzw. PISO auf die benötigten Iterationsschritte und die benötigte Rechenzeit aus? Sie können diese über das Menü `Solution > Solution Methods > Pressure-Velocity Coupling > Scheme` anwählen.
- Wie viele Iterationsschritte und welche Rechenzeit ist erforderlich, wenn Sie die Simulationen mit SIMPLE, SIMPLEC und PISO auch auf den beiden anderen im Praktikum *Ebenes, quadratisches Strömungsgebiet* generierten Gittern ausführen?
- Wie sieht das Ergebnis der Strömungssimulation aus, wie gut stimmen Ihre Ergebnisse mit bekannten Daten überein?

Diskussion Der Verlauf der Residuen R_u und R_v der u - und der v -Gleichungen in den Simulationen mit SIMPLE, SIMPLEC und PISO ist in Abb. 5.12 dargestellt ². Es ergibt sich der typische Verlauf für eine stabile iterative Lösung des Gleichungssystems. Die Residuen nehmen im Verlauf der Iteration ab und unterschreiten schließlich das vorgegebene Abbruchkriterium. Die Iteration wird dann abgebrochen und das Ergebnis der Simulation steht fest. Die Verläufe der Residuen bei den drei eingesetzten Druckkorrektur-Verfahren unterscheiden sich insgesamt nur gering, die Verläufe der Residuen von SIMPLEC und PISO sind sogar fast identisch.

Allerdings zeigen sich merkbare Unterschiede bei der Zahl der bis zum Iterationsabbruch notwendigen Iterationsschritte n_{Iter} und in der benötigten Rechenzeit t_{comp} ,

² Hinweis: Falls Sie ähnliche grafische Darstellungen der Residuenverläufe generieren wollen, können Sie über das Menü `File > Write > Start Transcript` eine Datei anlegen, in der die Konsolen-Ausgaben während der CFD-Simulation gespeichert werden. Der Inhalt der Datei kann anschließend nach ein wenig Vorbereitung beispielsweise mit gnuplot dargestellt werden.

Tab. 5.3 Nischenströmung mit OpenFOAM: Iterationsschritte n_{Iter} und Rechenzeiten t_{comp}

Gitter	SIMPLE		SIMPLEC		PISO	
	n_{Iter}	t_{comp}	n_{Iter}	t_{comp}	n_{Iter}	t_{comp}
50 × 50 Zellen	434	8 s	552	10 s	479	10 s
100 × 100 Zellen	956	34 s	1159	43 s	1045	44 s
200 × 200 Zellen	2429	305 s	2973	382 s	2545	412 s

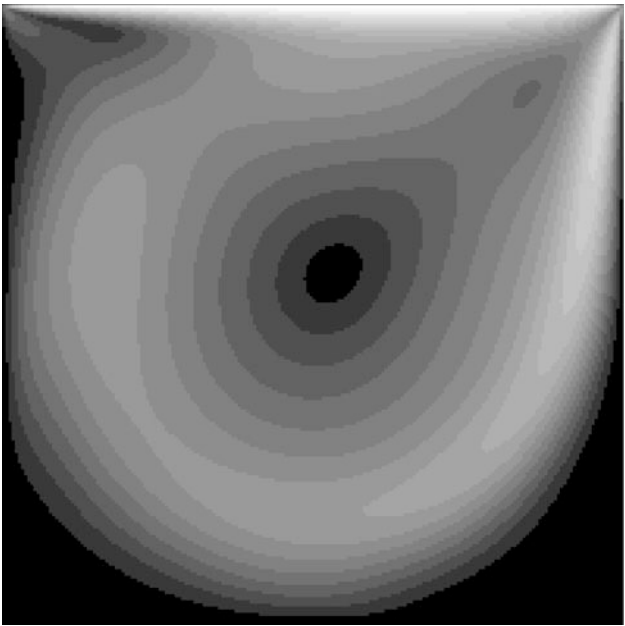
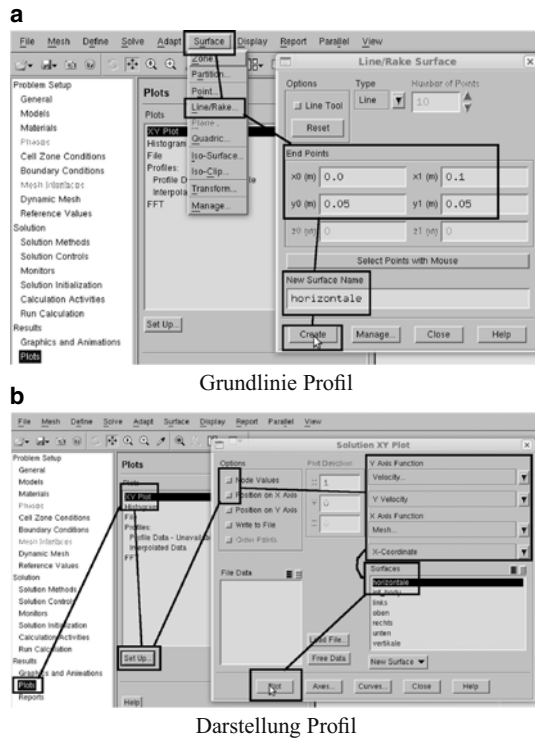


Abb. 5.13 Nischenströmung mit ANSYS FLUENT: Geschwindigkeitsfeld

Tab. 5.3. SIMPLE benötigt auf jedem Gitter weniger Iterationsschritte und auch weniger Rechenzeit bis zum Iterationsabbruch als SIMPLEC oder PISO. Dieser Befund bestätigt den generellen Trend, dass in modernen CFD-Programmen überwiegend der SIMPLE-Algorithmus zur Simulation von stationären Strömungsproblemen genutzt wird. Andere Druckkorrektur-Verfahren liefern in vielen Fällen keine schnellere Lösung. Allerdings kann es bei bestimmten Problemen durchaus von Vorteil sein, ein anderes Druckkorrektur-Verfahren zu erproben, um die Konvergenz der Lösung zu beschleunigen und so Rechenzeiten zu verkürzen.

Das in der Simulation auf dem Gitter mit 200 × 200 Zellen ermittelte Geschwindigkeitsfeld besitzt die erwartete Struktur, Abb. 5.13. Es existiert ein großes Rezirkulationsgebiet in der Nische, dabei werden die höchsten Geschwindigkeiten direkt an der bewegten oberen Wand gefunden.

Abb. 5.14 Nischenströmung mit ANSYS FLUENT: Auswertung Simulationsergebnisse



Grundlinie Profil

Darstellung Profil

Um Ihre Ergebnisse auch quantitativ mit den Daten von Ghia et al. [18] vergleichen zu können, definieren Sie im Menü Surface > Line/Rake jeweils eine vertikale und eine horizontale Profillinie durch den Mittelpunkt des Volumens, Abb. 5.14a. Anschließend kann der Verlauf einer Geschwindigkeitskomponente entlang dieser Profillinien (im Beispiel entlang der oben definierten Profillinie horizontale) dargestellt werden, dazu wählen Sie das Menü Results > Plots > XY Plots und wählen dort die entsprechenden Größen für die Darstellung aus. Verzichten Sie auf die glättende Darstellung der Daten, indem Sie die Option Node Values abwählen. Stellen Sie außerdem die Koordinaten entlang der Profillinie dar, hierzu müssen Sie die Option Position on X Axis abwählen und dann eine passende Auswahl unter X Axis Function treffen, Abb. 5.14b.

In Abb. 5.15 sind das berechnete Geschwindigkeitsprofil $u_y(x, y = 0,05 \text{ mm})$ (ANSYS FLUENT) und die entsprechenden Referenzwerte aus Ghia et al. [18] dargestellt. Die Übereinstimmung zwischen den Ergebnissen aus ANSYS FLUENT und den Referenzwerten ist gut, nur im Bereich des Geschwindigkeitsmaximums sind leichte Abweichungen zu erkennen. Vergleichen Sie auch die Profile aus den Simulationen auf den anderen Gittern mit den Referenzwerten.

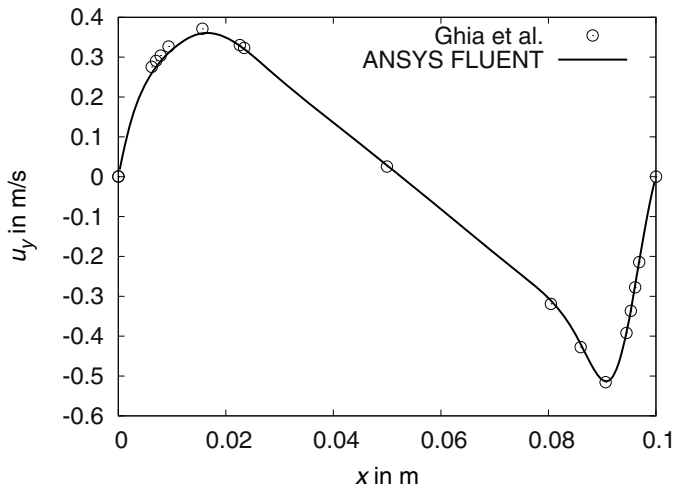


Abb. 5.15 Nischenströmung mit ANSYS FLUENT: Geschwindigkeitsprofil u_y ($x, y = 0,05$ mm)

5.3.3 Fazit

Die CFD-Simulation der Nischenströmung zeigt eine sehr gute qualitative und quantitative Übereinstimmung zwischen den berechneten Geschwindigkeitsprofilen und den Resultaten der Referenzuntersuchungen. Der in den Geschwindigkeitsprofilen gefundene Modellfehler ist sehr gering, da keine wesentlichen vereinfachenden Annahmen getroffen wurden. Im Vergleich der verschiedenen numerischen Löser zeigt sich, dass das Multigrid-Verfahren schneller konvergiert als das Gradienten-Verfahren (Simulationen mit Open-FOAM). Dagegen gibt es beim Vergleich der verschiedenen Druckkorrektur-Verfahren kaum Unterschiede im Rechenaufwand für die Rechnungen mit SIMPLE, SIMPLEC und PISO (Simulationen mit ANSYS FLUENT). Deswegen wird in CFD-Simulationen von stationären Strömungen in der Regel SIMPLE genutzt. Bei transienten Strömungen wird dagegen auch PISO recht häufig eingesetzt.

CFD-Modellierung

Grundlagen und Anwendungen bei
Strömungsprozessen

Schwarze, R.

2013, XII, 193 S. 176 Abb., 11 Abb. in Farbe. Mit

Online-Extras., Softcover

ISBN: 978-3-642-24377-6