

# GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN

NUMERISCHE STRÖMUNGSMECHANIK

---

## Simulation mit FTCS- und BTCS-Verfahren

---

*Author:*  
Philip MARSZAL

*Prüfer:*  
Prof. Dr. Andreas  
TILGNER

4. August 2016



## Zusammenfassung

Your abstract.

## 1 Einleitung

## 2 Problemstellung

$$\frac{\partial T(\mathbf{x}, t)}{\partial t} + \text{Pe } \mathbf{u}_0 \cdot \nabla T(\mathbf{x}, t) = \nabla^2 T(\mathbf{x}, t). \quad (1)$$

$$\frac{\partial T(\mathbf{x}, t)}{\partial t} + \text{Pe } \mathbf{u}_0 \cdot \nabla T(\mathbf{x}, t) = \nabla^2 T(\mathbf{x}, t) + Q(x, y). \quad (2)$$

## 3 Methoden

Zur Simulation von Strömungsmechanikproblemen kommen eine Vielzahl verschiedener Verfahren in Frage. Ein konzeptionell simples Verfahren, das häufig Anwendung findet, ist die *Finite-Differenzen-Methode*. Diese wird zum Lösen des hier gestellten Problems verwendet. Andere Verfahren die zur Lösung dieser Art von Problem eingesetzt werden können sind *Finite-Volumen-Methoden*, *Finite-Elemente-Verfahren* und *spektrale Verfahren*. Bei fast allen Verfahren liegt die Diskretisierung des integrierten Systems durch ein Gitter zu Grunde, welches die theoretisch kontinuierliche Lösungsfunktion approximiert.

### 3.1 Finite-Differenzen-Verfahren

Um eine partielle Differentialgleichung numerisch zu integrieren, müssen die Ableitungen ersetzt werden durch diskret berechenbare Terme. Ein naiver Ansatz ist die Näherung der Ableitung  $T'(x)$  einer Funktion an einem Punkt durch den Differenzenquotienten.

$$T'(x) = \lim_{h \rightarrow 0} \frac{T(x+h) - T(x)}{h}. \quad (3)$$

Bei der Verwendung eines Gitters entspricht hier  $h$  dem Abstand der zwei benachbarten Gitterpunkte. Der Differenzenquotient in Gleichung (3) entspricht dem Vorwärtsdifferenzenquotienten erster Ordnung in  $h$ .

Eine Approximation höherer Ordnung erhält man z.B. durch den zentralen Differenzenquotienten

$$T'(x) = \frac{T(x+h) - T(x-h)}{2h} + O(h^2), \quad (4)$$

oder für die zweite Ableitung als Vorwärtsdifferenzenquotient von  $T'(x)$ :

$$T''(x) = \frac{T(x+h) - 2T(x) + T(x-h)}{h^2} + O(h^2). \quad (5)$$

Durch Kombinationen unterschiedlicher Differenzenquotienten können Näherungen höherer Ordnung erhalten werden.

### 3.2 FTCS-Variante

Verwendet man zur Diskretisierung der PDGL eine Vorwärtsdifferenz für die Zeitableitung und eine zentrale Differenz für die Ortsableitungen, so spricht man von einem *FTCS*-Verfahren (**F**orwards **T**ime **C**entral **S**pace). Diskretisiert man Gleichung (1) auf diese Weise so erhält man die Differenzengleichung:

$$\begin{aligned} \frac{T_{i,j}^{n+1} - T_{i,j}^n}{\Delta t} + \text{Pe } u_{i,j} \frac{T_{i+1,j}^n - T_{i-1,j}^n}{\Delta x} + \text{Pe } v_{i,j} \frac{T_{i,j+1}^n - T_{i,j-1}^n}{\Delta y} \\ = \frac{T_{i+1,j}^n - 2T_{i,j}^n + T_{i-1,j}^n}{\Delta x^2} + \frac{T_{i,j+1}^n - 2T_{i,j}^n + T_{i,j-1}^n}{\Delta y^2} \end{aligned} \quad (6)$$

Diese Gleichung kann nun numerisch integriert werden, indem  $T_{i,j}^{n+1}$  für jeden Gitterpunkt berechnet wird. Allerdings ist darauf zu achten ob man  $T_{i,j}^{n+1}$  am Rand berechnet, denn hier müssen die Randbedingungen in die Gleichung eingebaut werden.

Die Dirichlet Randbedingungen lassen sich unproblematisch für  $T_{i,0}^n$  und  $T_{i,N_y+1}^n$  einsetzen. Diese Gitterpunkte verändern sich nicht im Laufe der Zeit, müssen also nicht integriert werden. Anders sieht dies allerdings für die Neumann Randbedingungen aus. Die Gitterpunkte mit  $i = 0$  und  $i = N_x + 1$  verändern sich sehr wohl im Laufe der Zeit.

Um zu vermeiden, dass man auf Gitterpunkte außerhalb des Gitters zugreift, muss man eine Vorwärtsdifferenz verwenden. Eine Näherung zweiter Ordnung für die Ableitung  $\frac{\partial T(x,t)}{\partial x}$  bei  $x = 0$  und  $x = 1$  lässt sich mit Gleichung (7)

berechnen:

$$\begin{aligned} T_{0,j}^{n+1} &= -\frac{1}{3} (T_{2,j}^{n+1} - 4T_{1,j}^{n+1}), \\ T_{N_x+1,j}^{n+1} &= -\frac{1}{3} (T_{N_x-1,j}^{n+1} - 4T_{N_x,j}^{n+1}). \end{aligned} \quad (7)$$

Mit Gleichung (6) und Gleichung (7) ist man nun in der Lage das Problem zu simulieren.

### 3.3 Die BTCS-Variante

Im Gegensatz zu dem FTCS-Verfahren, welches ein explizites Verfahren zu Lösung von partiellen Differentialgleichungen darstellt, handelt es sich bei der *BTCS*-Methode (**B**ackwards **T**ime **C**entral **S**pace) um ein implizites Verfahren. Implizite Verfahren erfordern die Lösung eines Gleichungssystems ehe ein Zeitschritt durchgeführt werden kann.

Im BTCS-Verfahren wird die Zeitableitung durch eine Rückwärtsdifferenz diskretisiert. Dadurch entsteht in dem vorliegenden Problem zunächst ein lineares Gleichungssystem aus  $(N_x + 1) \cdot (N_y + 1)$  Gleichungen für die  $T_{i,j}^n$ .

$$\begin{aligned} \frac{T_{i,j}^n - T_{i,j}^{n-1}}{\Delta t} + \text{Pe } u_{i,j} \frac{T_{i+1,j}^n - T_{i-1,j}^n}{\Delta x} + \text{Pe } v_{i,j} \frac{T_{i,j+1}^n - T_{i,j-1}^n}{\Delta y} \\ = \frac{T_{i+1,j}^n - 2T_{i,j}^n + T_{i-1,j}^n}{\Delta x^2} + \frac{T_{i,j+1}^n - 2T_{i,j}^n + T_{i,j-1}^n}{\Delta y^2} \end{aligned} \quad (8)$$

Führt man die Dirichlet-Randbedingungen ein so reduziert sich das Gleichungssystem auf  $N_x \cdot (N_y - 1)$  Gleichungen. Um die Neumann-Randbedingungen mit dem Gleichungssystem zu vereinigen betrachtet man den zentralen Differenzenquotienten an  $x = 0$  und  $x = 1$ :

$$\left. \frac{\partial T}{\partial x} \right|_{x=0} = \frac{T_{1,j}^n - T_{-1,j}^n}{\Delta x} = 0,$$

und einem analogen Quotienten für  $x = 1$ . Mithilfe dieser Gleichungen lassen sich die Terme  $T_{-1,j}^n$  und  $T_{N_x+2,j}^n$  aus dem Gleichungssystem eliminieren und somit die Neumann-Randbedingungen in die Gleichungen einbauen.

Um dieses System zu lösen bietet es sich an das Gitter  $T_{i,j}^n$  als  $N_x \cdot (N_y - 1)$  dimensionalen Vektor umzuschreiben. Die Lösung des Systems entspricht

dann einer Invertierung der Matrix  $M$ , die sich aus den Gleichungen als Gleichung (9) ergibt.

$$M = \begin{pmatrix} (1.+4D) & (v_{0,1}A-D) & 0 & \dots & -2D & 0 & \dots & \dots & 0 \\ -(v_{0,2}A+D) & (1.+4D) & (v_{0,2}A-D) & 0 & \dots & -2D & 0 & \dots & 0 \\ 0 & -(v_{0,3}A+D) & (1.+4D) & (v_{0,3}A-D) & 0 & \dots & \dots & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 0 \\ \dots & \dots & -(u_{i,j}A+D) & \dots & -(v_{i,j}A+D) & (1.+4D) & (v_{i,j}A-D) & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & (u_{i,j}A-D) & \dots \\ 0 & \dots & \dots & \dots & \dots & \dots & -2D & \dots & 0 \end{pmatrix} \quad (9)$$

Hierbei wurde der Vektor  $\mathbf{T}^n$  so gewählt, dass die Spalten des Gitters aneinander gehängt werden, sprich die Komponenten des Vektors laufen zunächst im  $j$ -Index durch :  $\mathbf{T}^n = (T_{0,1}^n, T_{0,2}^n, \dots, T_{0,N_y}^n, T_{1,1}^n, \dots, T_{N_x+1,N_y}^n)$ . Die Terme  $A = \frac{Pe \Delta t}{\Delta x}$  und  $D = \frac{\Delta t}{\Delta x^2}$  entsprechen den konstanten Vorfaktoren im Advektions- und Diffusionsterm.

### 3.4 Der SOR-Algorithmus

Die Matrix  $M$  hat für vernünftige Gittergrößen und Zeitschritte  $\Delta t$  die Eigenschaft, dass sie diagonal dominant ist. Also die Summe der Beträge der nicht-diagonalen Einträge in jeder Spalte nicht den Betrag des diagonalen Eintrags dieser Spalte übersteigt.

Für Matrizen mit dieser Eigenschaft liefert das sogenannte Gauß-Seidel-Verfahren eine approximative Lösung des Gleichungssystems  $M\mathbf{x} = \mathbf{b}$ , die garantiert gegen die tatsächliche Lösung konvergiert. Zunächst wird  $M$  in eine Summe aus untere und oberer Dreiecksmatrix,  $L$  und  $U$ , und der Diagonale  $D$  zerlegt. Die Iterationsvorschrift des Gauß-Seidel-Verfahrens sieht nun wie folgt aus:

$$\mathbf{x}_{n+1} = (L + D)^{-1}(\mathbf{b} - U\mathbf{x}_n) \quad (10)$$

Der Vorteil dieser Methode liegt darin, dass das Produkt von  $(L + D)^{-1}$  mit einem Vektor einfach über Vorwärtssubstitution berechnet werden kann, dank der Dreiecksform von  $L + D$ .

Eine Weiterentwicklung des Gauß-Seidel-Algorithmus ist die *successive over-relaxation*-Methode (SOR). Dazu wird Gleichung (10) zunächst durch den Korrekturvektor  $\mathbf{r}_n = M\mathbf{x}_n - \mathbf{b}$  ausgedrückt. Über den Relaxationsparameter  $\omega$  kann nun die Konvergenz des Gauß-Seidel-Verfahrens erheblich beschleunigt werden. Die Iterationsvorschrift des SOR-Algorithmus lautet:

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \omega(L + D)^{-1}(\mathbf{r}_n) \quad (11)$$

Für die Wärmetransportgleichung liegt der optimale Relaxationsparameter zwischen 0 und 2.

## 4 Programmstruktur

Zur Bearbeitung des Problems wurden die beiden Methoden, FTCS und BTCS, in zwei getrennten Programmen implementiert. Als interface mit den Programmen wurde eine Konfigurationsdatei erstellt, die es erlaubt Parameter der Algorithmen zu ändern, ohne das Programm neu kompilieren zu müssen. Die Interaktion mit der Konfigurationsdatei wurde in der Headerdatei *interface.h* implementiert, welche zudem sämtliche Funktionen zur Ausgabe der Ergebnisse beinhaltet.

Die Kernfunktionen der Integration befinden sich in der *integration.h*-Datei. Diese sind die Initialisierungen der Temperaturfelder, des Quellfeldes und der Geschwindigkeitsfelder. Außerdem sind einzelne Zeitschritte in den Funktionen *FTCS()* und *FTCS\_with\_Q()* zusammengefasst. Diese nehmen als Input das Temperaturfeld (call-by-reference), die Geschwindigkeitsfelder und das Quellfeld (call-by-value). Die Entscheidung, welche Gleichung integriert werden soll, wird in der Konfigurationsdatei getroffen.

Die BTCS-Methode benötigt weitere Funktionen, die von dem betrachteten Problem abhängen. Dies sind Funktionen zum transformieren des Feldes in einen Vektor und zurück (*reshape\_vector()* und *shape\_back()*), die Einführung der Dirichlet-Randbedingungen in den Vektor (*impose\_dirichlet()*) und die Initialisierung der Matrix  $M$  (*BCTS\_implicit\_Matrix()*).

Im BTCS Programm werden zudem algebraische Operationen benötigt, die in der Datei *operations.h* definiert sind. Dabei handelt es sich um standard Vektor-Matrix Operationen, aber auch die Triangularisierung der  $M$ -Matrix (*triangularize()*) und das von der Problemstellung unabhängige SOR-Verfahren (*SOR()*).

Innerhalb der Schleife über die Zeit wurde zudem eine Abfrage eingeführt, die zu in der Konfigurationsdatei definierten Zeitpunkten das aktuelle Temperaturfeld ausgibt.

Die Darstellung der Ergebnisse wurde mittels *Python-Matplotlib* durchgeführt.

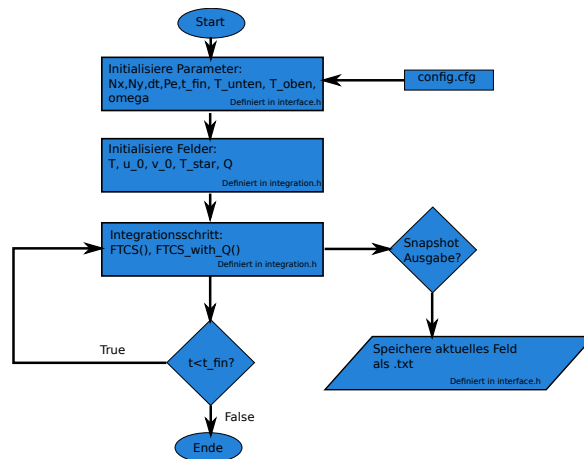


Abbildung 1: Flowchart der Implementierung der FTCS-Methode.

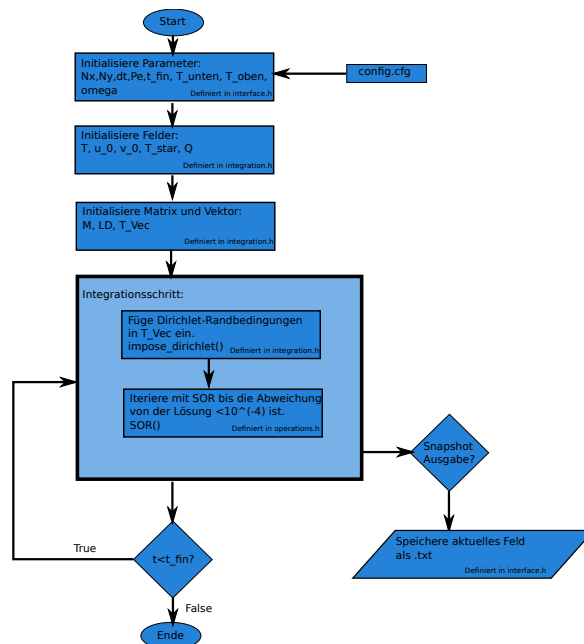


Abbildung 2: Flowchart der Implementierung der BTCS-Methode.

## 5 Ergebnisse

Die anfängliche Betrachtung des Temperaturfeldes im Verlauf der Zeit, wie es durch die FTCS-Methode berechnet wird, liefert die Ergebnisse in Abb. 6. Hiefür wurde eine Peclet-Zahl von  $Pe = 10$  und ein Gitter von  $N_x = N_y = 30$  angenommen. Die Schrittweite in der Zeit beträgt  $dt = 0.0002$ .

### 5.1 Fehler der FTCS-Lösung

Allerdings muss noch überprüft werden, wie genau das Ergebnis ist, und, wie die Gittergröße sich auf die Konvergenz auswirkt. Dazu wird ?? benutzt. Da die stabile stationäre Lösung dieser Gleichung bekannt ist verfügt man über ein Referenzfeld gegen das die FTCS-Näherung konvergieren müsste. In Abb. 5 ist die quadratische Abweichung von  $T^*$  als Funktion der Zeit für verschiedene Gittergrößen dargestellt. Es ist sichtbar, dass die numerische Lösung gegen einen konstanten Fehler konvergiert. Eine Verfeinerung des Gitters von  $N = 5$  auf 10 führt zu einer drastischen Verbesserung der Genauigkeit der Lösung, da der Fehler um fast eine Größenordnung sinkt. Bei weiterer Verfeinerung des Gitters kann ein weiterer Zuwachs an Genauigkeit festgestellt werden. ?? zeigt die Fehler zur Zeit  $t = 0.5$ . Sichtbar wird hier, dass der Zuwachs an Genauigkeit mit steigender Gittergröße abnimmt. Für die hier am häufigsten verwendete Gittergröße  $N = 30$  beträgt die *SSE* nach einer Zeit von 0.5 nur ca. 0.02. Eine SSE von nur 0.02 ist durchaus akzeptabel, vor allem da dies einer noch geringeren Abweichung der einzelnen Gitterpunkte von  $T^*$  entspricht, da die SSE nicht auf die Anzahl der Gitterpunkte normiert ist. Damit lässt sich sagen, dass das FTCS-Verfahren eine physikalisch sinnvolle Lösung des Problems liefert.

### 5.2 Vergleich unterschiedlicher Peclet-Zahlen

Nun da bekannt ist, dass die numerische Lösung des Verfahrens eine korrekte Lösung des Problems liefert, kann das ursprüngliche Problem aus Gleichung (1) genauer betrachtet werden. In ?? sind die Temperaturfelder für die Peclet-Zahlen  $Pe = 2$  und  $Pe = 10$  zu den Zeiten  $t = 0.005, 0.05$  und  $0.5$  visualisiert. Die Gittergröße beträgt wieder  $N = 30$  und die Zeitschrittweite  $dt = 0.0002$ .

Aus dem Vergleich wird deutlich welchen Einfluss die Peclet-Zahl auf die Dynamik des Systems hat. Eine größere Peclet-Zahl führt zu einer schnelleren



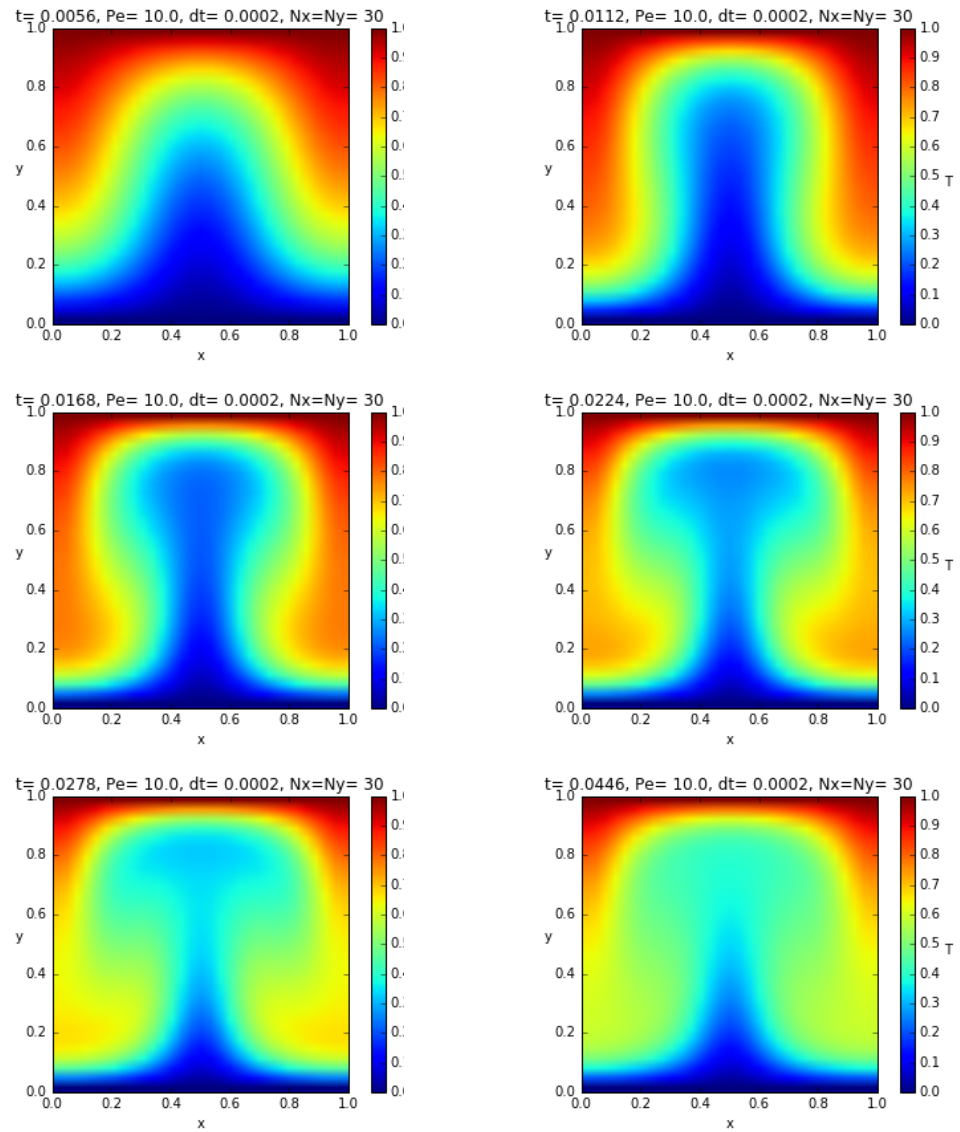


Abbildung 3: Temperaturfeld zu verschiedenen Zeiten. Berechnet mit dem FTCS-Verfahren.

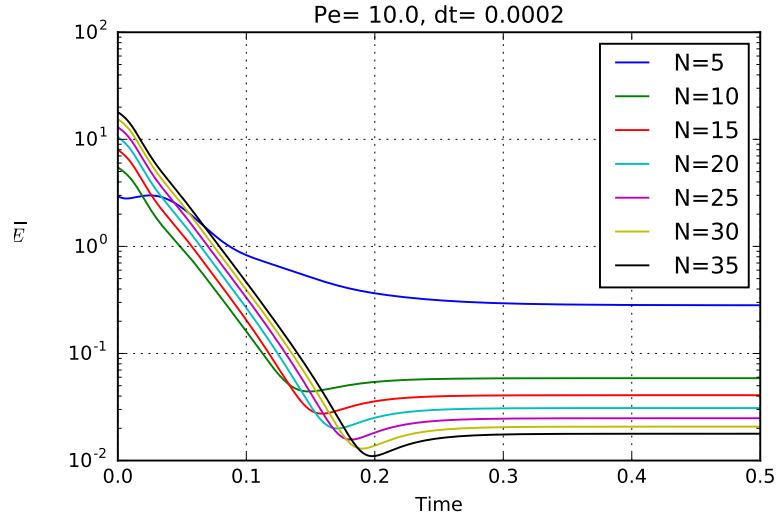


Abbildung 4: Sum of squared errors (SSE) der numerischen Integration im Vergleich zu dem Referenzfeld  $T^*$ .

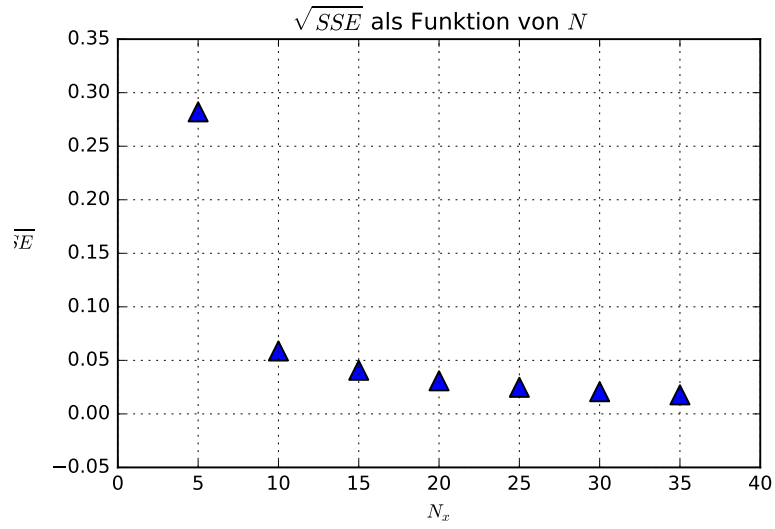


Abbildung 5: Sum of squared errors (SSE) der numerischen Integration im Vergleich zu dem Referenzfeld  $T^*$ .

advektiven Veränderung des Temperaturfeldes.

### 5.3 Zulässige Zeitschritte

Bis jetzt wurde immer der Zeitschritt  $dt = 0.0002$  verwendet. Aus den obigen Simulationen ging eindeutig hervor, dass für die betrachteten Gittergrößen und Peclet-Zahlen dieser Zeitschritt noch ein stabiles Verfahren liefert. Jetzt wird untersucht ab welcher Größe der Zeitschritt zu einer Divergenz des Verfahrens führt.

Eine theoretische Grenze für die Stabilität des Verfahrens wird durch das CFL-Kriterium für Diffusions- und Advektionsgleichungen vorgegeben. Für die Stabilität des Diffusionsterms muss folgende Ungleichung gelten:

$$\frac{\Delta t}{\Delta x^2} + \frac{\Delta t}{\Delta y^2} < CFL. \quad (12)$$

Wobei die CFL-Zahl für diese Ungleichung als 0.5 angenommen wird. Intuitiv lässt sich diese Gleichung verstehen als die Bedingung, dass die mittlere quadratische Verschiebung durch die Diffusion in einem Zeitschritt den Abstand zweier Gitterpunkte nicht übersteigen darf. Für eine Diffusionskonstante von 1 ergibt sich damit die CFL-Zahl 0.5.

Die zweite Stabilitätsbedingung des FTCS-Verfahrens rührt von dem Advektionsterm her und ist ähnlich zu verstehen wie Gleichung (12). In einem Zeitschritt darf die Verschiebung der Temperatur nur maximal den nächsten Gitterpunkt erreichen. Die Ungleichung für den zwei dimensional Fall lautet:

$$\frac{Pe \, u_{max} \Delta t}{\Delta x} + \frac{Pe \, v_{max} \Delta t}{\Delta y} < CFL = 1. \quad (13)$$

Für unser gegebenes Geschwindigkeitsfeld ergibt sich  $u_{max} = \pi$  und  $v_{max} = 2\pi$ . In Abb. 7 sind die Stabilitätsgrenzen als Funktionen der Peclet-Zahl dargestellt. Der Umschlag des jeweils geltenden Kriteriums befindet sich bei  $Pe = 12.56$  für eine konstant gehaltene Gittergröße mit  $N = 30$ .

Für die Peclet-Zahlen  $Pe = (0.1, 1, 10, 20, 100)$  wurde das System mit unterschiedlichen Zeitschritten integriert. Abb. 8 zeigt die Entwicklung des Mittelwertes der Beträge der Gitterpunkte. Die jeweils größten stabilen Zeitschritte sind in Abb. 7 als Punkte eingetragen.

Aus der Analyse von Abb. 8 geht hervor, dass für  $Pe = (0.1, 1, 10, 20)$  selbst Zeitschritte, die leicht über der Stabilitätsschranke liegen noch stabil erscheinen.

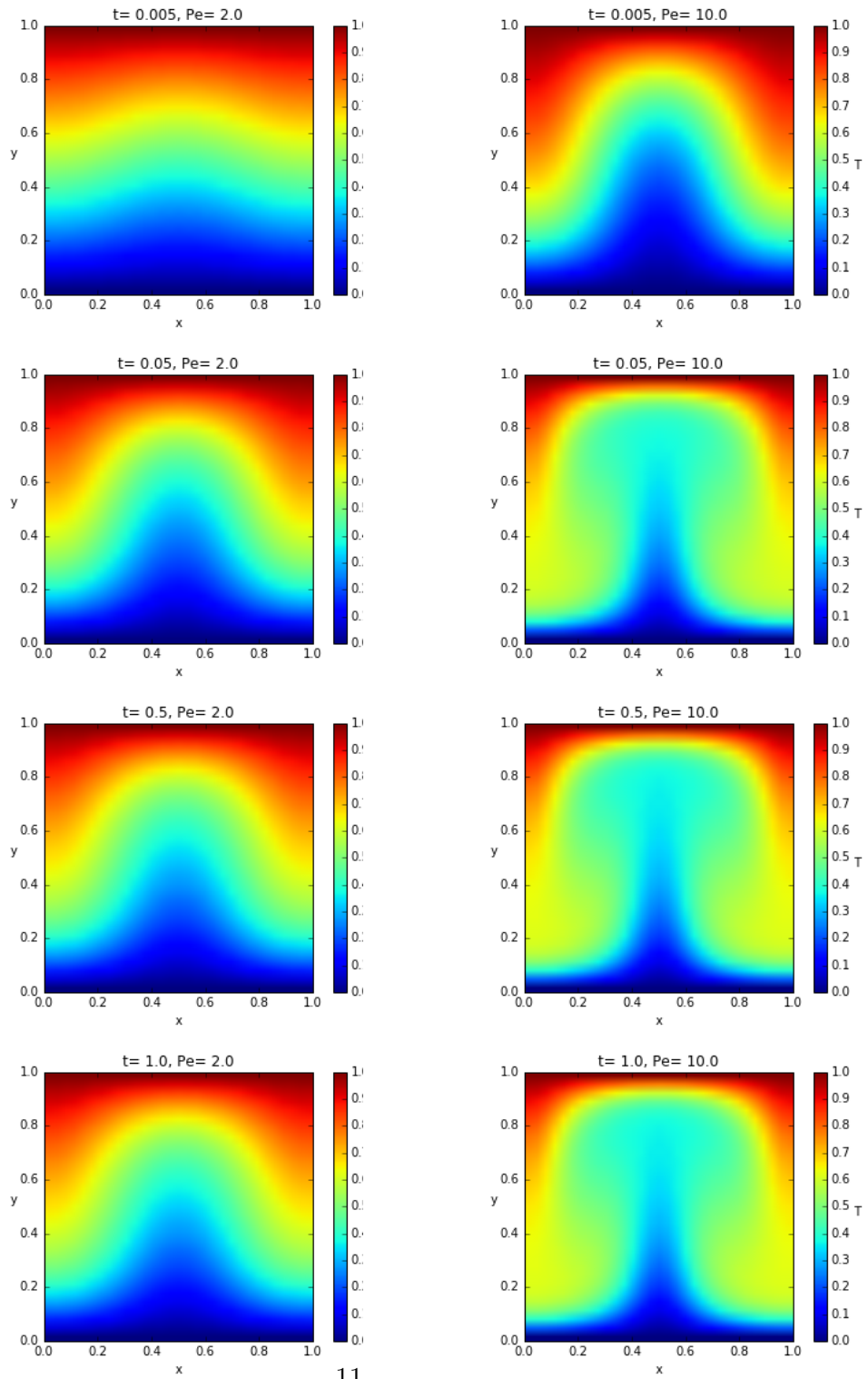


Abbildung 6: Temperaturfeld zu verschiedenen Zeiten und verschiedenen  $Pe$ . Berechnet mit dem FTCS-Verfahren. Links  $Pe = 2$ , rechts  $Pe = 10$ .

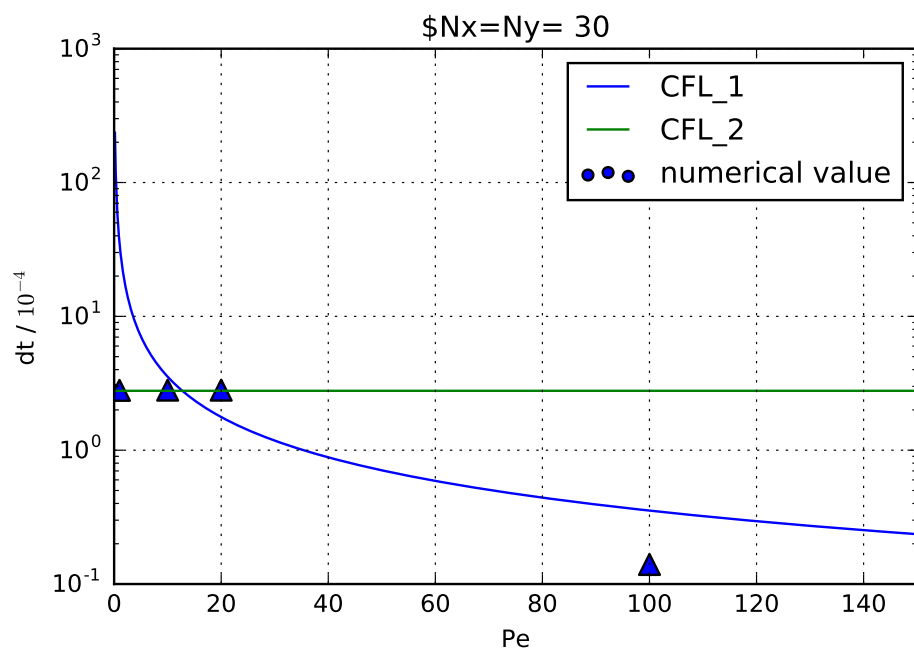


Abbildung 7: Theoretisch erwartete Grenzen für die Stabilität in Abhängigkeit von der Peclet-Zahl. Die Punkte markieren die in Abb. 8 gefundenen maximal stabilen Werte.

Allerdings geht aus den Simulationen hervor, dass die der Divergenz für kleinere Zeitschritte später einsetzt, bzw. erkennbar wird. Daraus kann man schließen, dass man bessere Näherungen für den maximalen stabilen Zeitschritt erhält wenn man den Integrationszeitraum erhöht.

Ein anderes Problem tut sich jedoch auf wenn man  $Pe = 100$  betrachtet. Diese wurde hier lediglich betrachtet, weil der Umschlag der Stabilitätsgrenze bei  $Pe = 20$  nicht erkennbar war, wahrscheinlich wegen des oben beschriebenen Phänomens.

Im Gegensatz zu den kleineren Peclet-Zahlen, liefert die numerische Untersuchung von  $Pe = 100$  Zeitschritte die nach Gleichung (13) stabil sein sollten, aber in der Praxis zu Divergenz führen.

Diese Tatsache verdeutlicht, dass es sich bei den CFL-Kriterien nur um notwendige Kriterien handelt und nicht um hinreichende. Eine Neumann'sche Stabilitätsanalyse könnte weitere Kriterien für die Stabilität des Verfahren für größere Peclet-Zahlen aufzeigen.

## 6 Vergleich FTCS- und BTCS

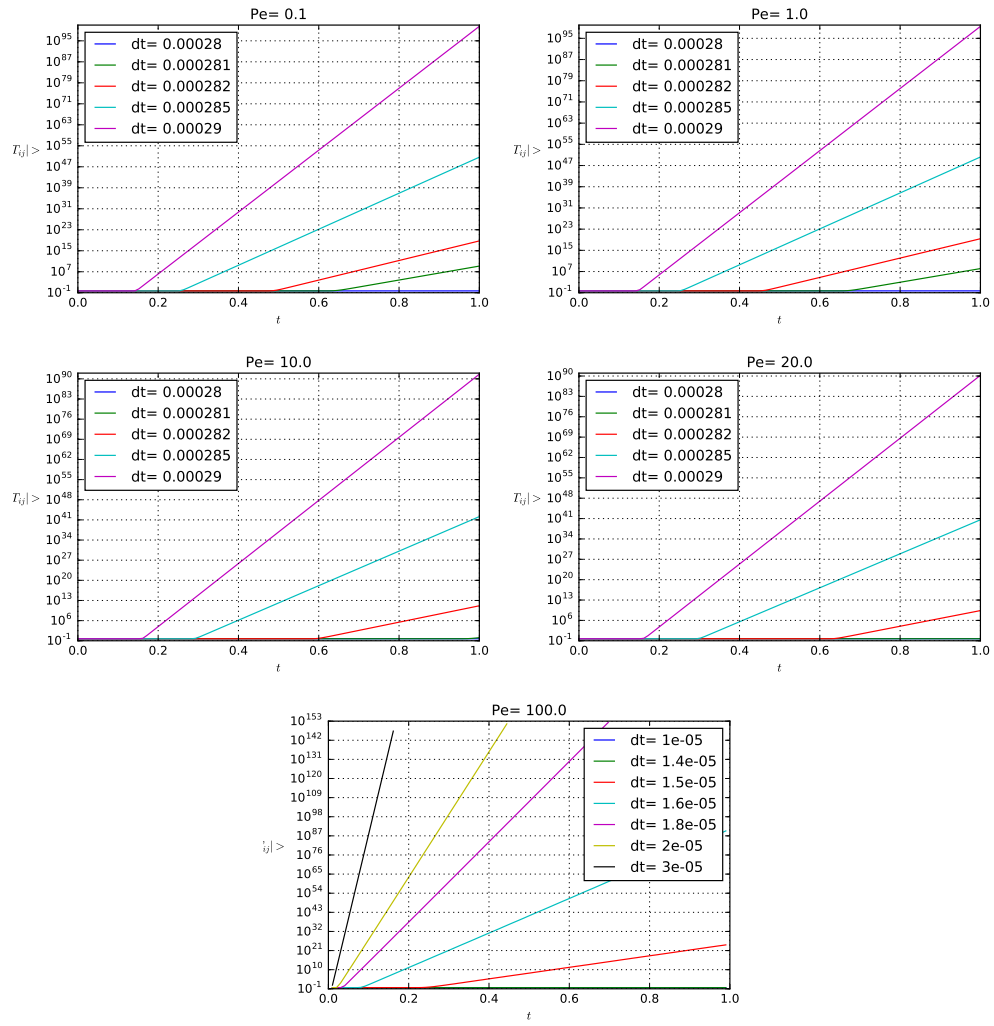


Abbildung 8: Suche nach dem maximalen stabilen Zeitschritt für verschiedene Peclet-Zahlen.