ASC Report No. 07/2015

Splittingverfahren für die Gray-Scott-Gleichung

W. Auzinger, O. Koch, M.Quell

Institute for Analysis and Scientific Computing —
Vienna University of Technology — TU Wien
www.asc.tuwien.ac.at ISBN 978-3-902627-05-6

Most recent ASC Reports

06/2015 *M. Kaltenbäck, R. Pruckner*
Functional Calculus for definitizable self-adjoint linear relations of Krein spaces

05/2015 *H. de Snoo and H. Woracek*
Restriction and factorization for isometric and symmetric operators in almost Pontryagin spaces

04/2015 *N. Zamponi and A. Jüngel*
Analysis of degenerate cross-diffusion population models with volume filling

03/2015 *C. Chainais-Hillairet, A. Jüngel, and P. Shpartko*
A finite-volume scheme for a spinorial matrix drift-diffusion model for semiconductors

02/2015 *T. Horger, J.M. Melenk, B. Wohlmuth*
On optimal L^2- and surface flux convergence in FEM (extended version)

01/2015 *W. Auzinger and W. Herfort*
An Application of Gröbner Bases to perturbed Polynomial Equations

42/2014 *H. Woracek*
Perturbation of chains of de Branges spaces

41/2014 *T. Führer, J.M. Melenk, D. Praetorius, A. Rieder*
Optimal additive Schwarz methods for the hp-BEM: the hypersingular integral operator in 3D on locally refined meshes

40/2014 *M. Miletić, D. Stürzer and A. Arnold*
An Euler-Bernoulli beam with nonlinear damping and a nonlinear spring at the tip

39/2014 *T. Horger, J.M. Melenk, B. Wolmuth*
On optimal L^2- and surface flux convergence in FEM

Institute for Analysis and Scientific Computing
Vienna University of Technology
Wiedner Hauptstraße 8–10
1040 Wien, Austria

E-Mail: admin@asc.tuwien.ac.at
WWW: http://www.asc.tuwien.ac.at
FAX: +43-1-58801-10196

ISBN 978-3-902627-05-6

ASC TU WIEN

Splittingverfahren für die Gray-Scott-Gleichung[1]

W. Auzinger, O. Koch, M. Quell

1 Einleitung

Das als Gray-Scott[2] bezeichnete System von partiellen Differentialgleichungen in drei Dimensionen, beschreibt wie gewisse Chemikalien u und v miteinander reagieren.

$$\begin{aligned}\frac{\partial u}{\partial t} &= D_u \Delta u - uv^2 + F(1-u)\\ \frac{\partial v}{\partial t} &= D_v \Delta v + uv^2 - (F+k)v\end{aligned}$$

Die Δ-Terme beschreiben die Diffusion mit den für die Chemikalien spezifischen Faktoren D_u und D_v. Die Hauptreaktion wird durch den nichtlinearen Term beschrieben: $u + 2v \to 3v$. Der letzte Term beschreibt das Hinzufügen von u in Abhängigkeit vom vorhanden u, während v mit konstantem Faktor $F + k$ entnommen wird.
Die Konstanten wurden wie folgt gewählt:

$$D_u = 0.04, D_v = 0.005, F = 0.038, k = 0.076$$

Die gewählten Anfangsbedingung für $t_0 = 0$ sind:

$$\begin{aligned}u(t_0, x, y, z) &= 0.5 + e^{-1-\left(x^2+y^2+z^2\right)}\\ v(t_0, x, y, z) &= 0.1 + e^{-1-\left(x^2+y^2+z^2\right)}\end{aligned}$$

Die Gleichung wird auf dem Würfel $[-4\pi, 4\pi]^3$ mit periodischen Randbedingungen gelöst.

2 Allgemeines

Das System besteht aus zwei parabolischen partiellen Differentialgleichungen 2. Ordnung. Zur Lösung wird zuerst der Ort mit einem äquidistanten Gitter diskretisiert.

2.1 Splitting Verfahren

Sei $u' = f(u) + g(u), u(0) = u_0$ eine Differentialgleichung mit $f, g \in C^0$. Splitting Verfahren sind „ Teile und Herrsche “ Verfahren, da man die Gleichung in zwei leichter zu lösende Subprobleme aufspaltet.

$$\begin{aligned}u'_f &= f(u_f), \quad u_f(0) = u_{f_0}\\ u'_g &= g(u_g), \quad u_g(0) = u_{g_0}\end{aligned}$$

In diesen Fall wäre die Lösung jeweils für Schrittweite h

$$\begin{aligned}u_f(h) &= \phi_f(h, u_{f_0})\\ u_g(h) &= \phi_g(h, u_{g_0})\end{aligned}$$

[1] Die numerischen Resultate wurden auf dem *Vienna Scientific Cluster* (VSC) erzielt.

Anschließendes verknüpfen der Teillösungen führt auf

$$u(h) \approx u_g(u_f(h)) = \phi_g(h, \phi_f(h, u_{f_0}))$$

Insbesondere im linearen Fall

$$u' = Au + Bu, \quad u(0) = u_0$$

mit exakter Lösung

$$u(h) = e^{h(A+B)}u_0$$

ist sofort ersichtlich, dass das nur mit

$$u(h) = e^{hA}e^{hB}u_0$$

übereinstimmt wenn A und B kommutieren. Dieses Verfahren ist auch als Lie-Splitting bekannt. Es hat die Splitting-Koeffizienten $a = 1$ und $b = 1$. Um Splitting-Verfahren höherer Ordnung zu konstruieren benötigt man mehr Koeffizienten. Das allgemeine Schema für $a_i, b_i \in \mathbb{C}$ unter den minimal Bedingungen $\sum_{i=1}^{s} a_i = 1$ und $\sum_{i=1}^{s} b_i = 1$ sieht wie folgt aus:

$$u(h) \approx e^{ha_1A}e^{hb_1B}e^{ha_2A}e^{hb_2B} \cdots e^{ha_sA}e^{hb_sB}u_0$$

Das Splitting in der Gray-Scott-Gleichung wird folgendermaßen durchgeführt. Die Gleichung wird in den nichtlinearen

$$\begin{aligned} \frac{\partial u}{\partial t} &= -uv^2 \\ \frac{\partial v}{\partial t} &= uv^2 \end{aligned}$$

und linearen Teil gesplittet.

$$\begin{aligned} \frac{\partial u}{\partial t} &= D_u\Delta u + F(1-u) \\ \frac{\partial v}{\partial t} &= D_v\Delta v - (F+k)v \end{aligned}$$

Der nichtlineare Teil wird mit der impliziten Mittelpunktsregel berechnet die mit einer Fixpunktiteration gelöst wird.
Der lineare Teil hingegen kann durch Wechsel in den Frequenzraum exakt berechnet werden, wobei k_x, k_y und k_z die entsprechenden Fourier Frequenzen sind. $\mathcal{F}$ die diskrete Fouriertransformation, die mittels [I]FFT realisiert wird.

$$\begin{aligned} \mathcal{F}(u(t)) &= e^{t(D_u(k_x^2+k_y^2+k_z^2)-F)}\left(D_u(k_x^2+k_y^2+k_z^2)\mathcal{F}(u_0) - \mathcal{F}(F)\right) + \frac{\mathcal{F}(F)}{D_u(k_x^2+k_y^2+k_z^2)-F} \\ \mathcal{F}(v(t)) &= e^{t(D_v(k_x^2+k_y^2+k_z^2)-F-k)}\mathcal{F}(v_0) \end{aligned}$$

2.2 Adaptive Integration

Um bei vorgegebener Fehlertoleranz die effizienteste Integration durchzuführen empfiehlt es sich die Schrittweite in jedem Schritt anzupassen. Dadurch kann erreicht werden, das jeder einzelne Zeitschritt maximale Länge hat. Es entsteht ein neues Problem, wie errechnet man die Größe des Fehlers im nächsten Zeitschritt und wählt die nächste Schrittweite. Die Lösung ist den Fehler in einem Schritt möglichst effizient zu schätzen und dann die Schrittweite im nächsten Schritt entsprechend zu errechnen.

2.3 Fehler- und Schrittweitenbestimmung

[7, pp. 62-64][3] Der Fehler im n-ten Schritt wird durch den Vergleich von zwei Verfahren unterschiedlicher Ordnung geschätzt. In der Implementierung wurde die Maximums Norm verwendet.

$$err_n = \|u_n - \hat{u}_n\| + \|v_n - \hat{v}_n\|$$

Die Approximation niedriger Ordnung wird durch $\hat{}$ gekennzeichnet. In Abhängigkeit von der Ordnung p des Verwendeten Verfahrens mit der höheren Ordnung, vom geschätzten Fehler err_n und der vorgegeben Toleranz tol wird die optimale Schrittweite h_{n+1} für den nächsten Zeitschritt bestimmt:

$$h_{n+1} = \min\left(h_{\max}, \max\left(h_{\min}, h_n \sqrt[p+1]{\alpha \frac{tol}{err_n}}\right)\right)$$

Dabei hat α den Wert 0.9, um ein Überschätzen der Schrittweite zu vermeiden, da sonst der Zeitschritt wiederholt werden müsste. Die maximalen Schrittweite wurden so gewählt, dass pro Ausgabe mindesten ein Zeitschritt gemacht wird. Die minimale Schrittweite wurde auf die Fehlertoleranz gesetzt.

2.4 Ordnung

Numerische Bestimmung der Ordnung eines Verfahrens. Sei u die exakte Lösung und u_h die mit der Schrittweite h berechnete Approximation, dann lässt sich der Fehler für drei aufeinanderfolgende Schrittweiten wie folgt abschätzen. p soll die Ordnung des verwendeten Verfahrens sein.

$$\begin{aligned}
\|u_h - u\| &= e_1 h^p + \dots \\
\left\|u_{\frac{h}{2}} - u\right\| &= e_1 \left(\frac{h}{2}\right)^p + \dots \\
\left\|u_{\frac{h}{4}} - u\right\| &= e_1 \left(\frac{h}{4}\right)^p + \dots
\end{aligned}$$

Durch Differenzenbildung erhält man

$$\begin{aligned}
\left\|u_h - u_{\frac{h}{2}}\right\| &= e_1 h^p \left(1 - \frac{1}{2^p}\right) + \dots \\
\left\|u_{\frac{h}{2}} - u_{\frac{h}{4}}\right\| &= e_1 \left(\frac{h}{2}\right)^p \left(1 - \frac{1}{2^p}\right) + \dots
\end{aligned}$$

Anschließend wird dividiert und logarithmiert.

$$\frac{\left\|u_h - u_{\frac{h}{2}}\right\|}{\left\|u_{\frac{h}{2}} - u_{\frac{h}{4}}\right\|} = 2^p \Rightarrow p = \log_2 \frac{\left\|u_h - u_{\frac{h}{2}}\right\|}{\left\|u_{\frac{h}{2}} - u_{\frac{h}{4}}\right\|}$$

3 Lösen mit Strang(Lie–Trotter) Splitting 2(1)

Splitting Koeffizienten für Simple Verfahren:

Lie-Trotter Splitting

j	$\hat{a}_j$	$\hat{b}_j$
1	0.0	1.0
2	1.0	0.0

Strang Splitting

j	a_j	b_j
1	$\hat{a}_1$	0.5
2	1.0	0.5

Folgende Tabelle zeigt die numerischen Ergebnisse der Ordnungsbestimmung. Der Fehler ist Definiert als $err := \|u_{err}\|_\infty + \|v_{err}\|_\infty$ wobei $u_{err} := u_{approx} - u_{exakt}$ und analog für v_{err}. u_{approx} ist die errechnete Approximation an die exakte Lösung u_{exakt}. In diesem Fall ist das, die mit der Schrittweite h errechnete Approximation und die mit $h/2$ berechnete exakte Lösung. Da die Toleranz bei der Fixpunktinteration bei 10^{-12} angesetzt war, hat der Fehler eben diese untere Schranke. Die Ordnung wird mit den weiter oben vorgestellten Verfahren errechnet. Es wurde über das Intervall $t \in [0, 1]$ integriert.

Schrittweite	Fehler	Ordnung
$1.0000000E+00$	$1.8317615E-03$	1.77
$5.0000000E-01$	$5.3630690E-04$	1.94
$2.5000000E-01$	$1.3938564E-04$	1.98
$1.2500000E-01$	$3.5185181E-05$	1.99
$6.2500000E-02$	$8.8175794E-06$	1.99
$3.1250000E-02$	$2.2057266E-06$	1.99
$1.5625000E-02$	$5.5151445E-07$	1.99
$7.8125000E-03$	$1.3788285E-07$	2.00
$3.9062500E-03$	$3.4469075E-08$	2.00
$1.9531250E-03$	$8.6134143E-09$	2.00
$9.7656250E-04$	$2.1455818E-09$	2.04
$4.8828125E-04$	$5.2103286E-10$	−1.28

4 Lösen mit eingebetteten Verfahren 4(3)

Die folgende Tabelle zeigt Splitting-Koeffizienten für das verwendete eingebettete Verfahren der Ordnung 4(3).

Verfahren 4. Ordnung

j	a_j	b_j
1	0	$0.1621982020100856 + 0.0672931362454034i$
2	$0.3243964040201712 + 0.1345862724908067i$	$0.3378017979899144 - 0.0672931362454034i$
3	$0.3512071919596576 - 0.2691725449816134i$	$0.3378017979899144 - 0.0672931362454034i$
4	$0.3243964040201712 + 0.1345862724908067i$	$0.1621982020100856 + 0.0672931362454034i$

[4]

Verfahren 3. Ordnung

j	$\hat{a}_j$	$\hat{b}_j$
1	a_1	b_1
2	$0.4157701540561051 + 0.2129482257474245i$	$0.4052251807333103 + 0.1988642124619028i$
3	$0.3855092282056243 - 0.1105557092016989i$	$0.4325766172566041 - 0.2661573487073062i$
4	$0.1987206177382706 - 0.1023925165457255i$	0

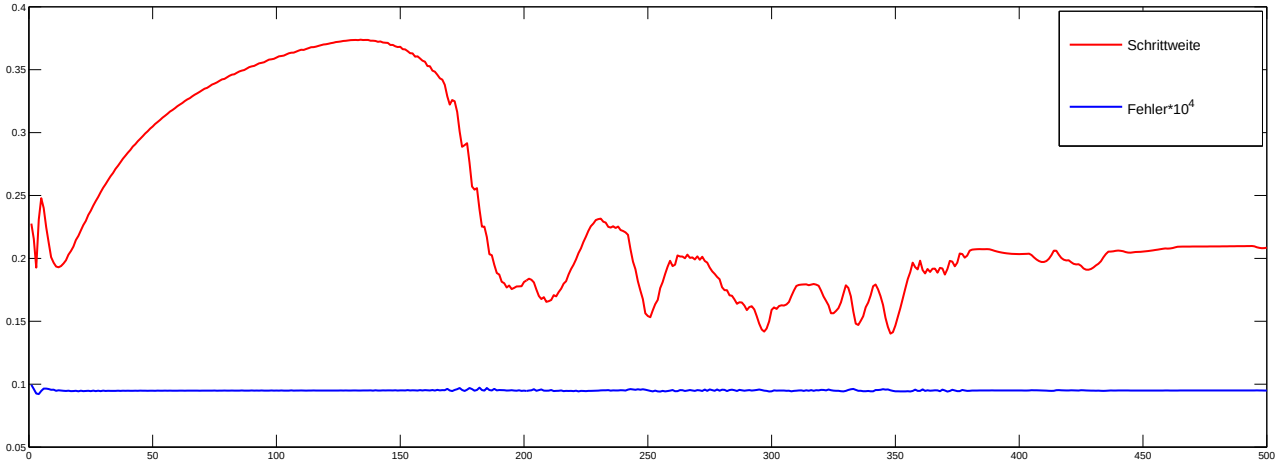


Abbildung 1: Ergebnis einer Simulation für Schrittweite und lokalen Fehler in Abhängigkeit von der Zeit t auf einem Gitter von 512^3 und einer Fehlertoleranz von 10^{-5} und oben festgesetzten Parametern für die Konstanten und Anfangsbedingung. Es ist deutlich zu erkennen das durch die adaptive Schrittweitenbestimmung mehr als doppelt so große Schritte möglichen waren, als wenn man alles mit fester Schrittweite gerechnet hätte und trotzdem die Fehlertoleranz eingehalten hätte.

Folgende Tabelle zeigt die Ergebnisse der numerischen Ordnungsbestimmung für das Eingebettete Verfahren. Der Fehler der Approximation mit Schrittweite h wird immer auf die numerisch errechnete Lösung mit Schrittweite $h/2$ bezogen. Die Toleranz bei der Fixpunktinteration ist bei 10^{-12} angesetzt, hat der Fehler eben diese untere Schranke. Die Ordnung wird mit den weiter oben vorgestellten Verfahren errechnet. Das Integrationsintervall ist $t \in [0, 1]$.

Schrittweite	Fehler	Ordnung
$1.0000000E+00$	$8.2691839E-05$	3.85
$5.0000000E-01$	$5.7072105E-06$	3.96
$2.5000000E-01$	$3.6595134E-07$	3.99
$1.2500000E-01$	$2.3019260E-08$	3.99
$6.2500000E-02$	$1.4407451E-09$	4.00
$3.1250000E-02$	$8.9852473E-11$	2.33
$1.5625000E-02$	$1.7789522E-11$	-1.00

5 Verwendete Rechner

Der **Vienna Scientific Cluster 2**[8] ist ein High Performance Computing (HPC) Cluster. Er besteht aus 1314 Rechenknoten, die mit je 2 AMD Opteron („Magny Cours 6132HE“ je 8 Kerne und 2,2 GHz Taktfrequenz), 8 x 4 GB ECC DDR3 RAM Arbeitsspeicher, 16 GB SSD Festplatte, 2 x Gigabit Ethernet LAN und 1 x Infiniband-QDR ausgerüstet sind. Zusätzlich gibt es noch 3 Zugangsknoten und 2 Masterknoten, die jedoch eine 300 GB Festplatte anstatt der SSD verbaut haben und über redundante Netzteile verfügen. Die Daten werden auf den 12 Storageserver mit temporären Dateisystem (FHGFS) und 2 NFS-Server gespeichert. Die Server haben je 2 64 Bit Intel Xeon Westmere E5620 (Quadcore), 6x8 GB ECC DDR3 RAM, 5x300 GB SAS für Betriebssystem, 12x2 TB SAS für Nutzdaten (FHGFS), 24x2 TB (NFS), ebenfalls mit redundantem Netzteil. Das Betriebssystem ist Scientific Linux 6.0 mit Intel MPI und Open MPI. Als Compiler stehen der „Intel Fortran & C Compilers (Composer XE 2011)“ und „GNU Fortran Compiler“ zur Verfügung.

6 Source Code

Der Source Code ist in Fortran geschrieben und verwendet als zusätzliche Bibliotheken MPI und „2DECOMP&FFT“. [5] MPI wird für die Kommunikation zwischen den einzelnen Prozessen genutzt. „2DECOMP&FFT“ wird für die Aufteilung der Datenfelder verwendet, da der Code auf einem System mit getrennten Speicherbereichen laufen soll. Die FFT ist die einzige Operation, die nicht lokal ausgeführt werden kann und wird somit auch von der Softwarebibliothek übernommen.

Im Programm werden zuerst alle Variablen deklariert und MPI initialisiert. Anschließend werden Variablen wie Gitterauflösung, Integrationsintervall und die Variablen der Gleichung aus der Datei „INPUTFILEa“ eingelesen. Sodann wird die „2DECOMP&FFT“ Bibliothek initialisiert und der Speicher allokiert. Jetzt können die Datenfelder mit den Anfangsdaten befüllt werden. Sobald dieser Vorgang abgeschlossen ist wird, die Zeitmessung und die Integration gestartet.

6.1 Lösung des linearen Teils

```
!uhat, vhat :: Fourier transformierten von u und v
!myid       :: Nummer des Prozesses
!dt         :: Schrittgroesse
!bhigh      :: Splittingkoeffizient
!decomp     :: aus 2DECOMP FFT speichert Anfang und Ende der Felder
!A          :: F aus der Gleichung
!B          :: F+k aus der Gleichung
```

```
!Du, Dv      :: D_u und D_v aus der Gleichung
!Ahat        :: Da die Fourier transformierte vom diskretisierten Operator F nur an
!            (1,1,1) nicht null ist. Um Speicher zu sparen wird nur dieser Eintrag
!            gespeichert und dem Punkt (1,1,1) eine Sonderbehandlung zu teil.

    IF (myid.eq.0) THEN
        uhatmp=uhat(1,1,1)
    END IF
    DO k=decomp%zst(3),decomp%zen(3)
    DO j=decomp%zst(2),decomp%zen(2)
        DO i=decomp%zst(1),decomp%zen(1)
            uhat(i,j,k)=exp(dt*bhigh(l)*(-A+Du*(kx(i)+ky(j)+kz(k))))*&
                uhat(i,j,k)
            vhat(i,j,k)=exp(dt*bhigh(l)*(-B+Dv*(kx(i)+ky(j)+kz(k))))*&
                vhat(i,j,k)
            END DO
        END DO
    END DO
    IF (myid.eq.0) THEN
        uhat(1,1,1)=exp(dt*bhigh(l)*(-A+Du*(kx(1)+ky(1)+kz(1))))*&
            (uhatmp-Ahat/(A+Du*(kx(1)+ky(1)+kz(1))))+&
            (Ahat/(A+Du*(kx(1)+ky(1)+kz(1))))
    END IF
```

6.2 Lösung des nichtlinearen Teils

```
!u, v        :: u und v
!uold, vold  :: temporaere Variable fuer die Fixpunktiteration
!utmp, vtmp  :: temporaere Variable fuer die Fixpunktiteration
!umean,vmean:: temporaere Variable fuer die Fixpunktiteration
!dt          :: Schrittgroesse
!decomp      :: aus 2DECOMP FFT speichert Anfang und Ende der Felder
!ahigh       :: Splittingkoeffizient
!chg         :: aenderung in der Fixpunktiteration
!tol         :: Abbruchbedingung an die Fixpunktiteration

    DO k=decomp%xst(3),decomp%xen(3)
        DO j=decomp%xst(2),decomp%xen(2)
            DO i=decomp%xst(1),decomp%xen(1)
                uold=u(i,j,k)
                vold=v(i,j,k)
                chg=1.0
                DO WHILE (chg>tol)
                    utmp=u(i,j,k)
                    vtmp=v(i,j,k)
                    umean=0.5*(u(i,j,k)+uold)
                    vmean=0.5*(v(i,j,k)+vold)
                    u(i,j,k)=uold+dt*ahigh(l)*(-umean*vmean**2)
                    v(i,j,k)=vold+dt*ahigh(l)*(umean*vmean**2)
                    chg=abs((u(i,j,k)-utemp))+&
                        abs((v(i,j,k)-vtemp))
                END DO
            END DO
        END DO
    END DO
```

Nachdem die beiden Zeitschritte verschiedener Ordnung abgearbeitet wurden wird der Fehler geschätzt und die Schrittweite angepasst. Sollte jedoch der Fehler die Toleranz überschreiten,

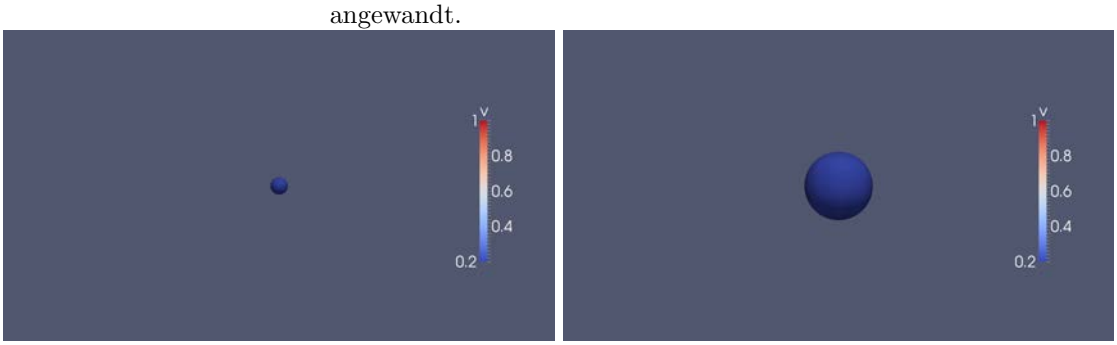
wird der Schritt mit der angepassten Schrittweite wiederholt. Muss ein Schritt mehr als 10 mal wiederholt werden, bricht das Programm ab.
Wird ein Schritt angenommen, dann wird geprüft ob der jetzige Stand der Simulation abgespeichert werden soll. Durch das Einlesen des „INPUTFILEa“ werden die Zeiten festgelegt an denen eine Ausgabe erwartet wird. Wird diese Zeit zum ersten mal überschritten wird eine Ausgabe eingeleitet. Dadurch liegt die Ausgabe maximal um $h_{\max}$ daneben.
Wenn die Endzeit erreicht ist, wird die Zeitmessung gestoppt, die Schrittweiten- und Fehlervektoren gespeichert und alle Felder deallokiert. Dann terminiert das Programm.

7 Visualisierung

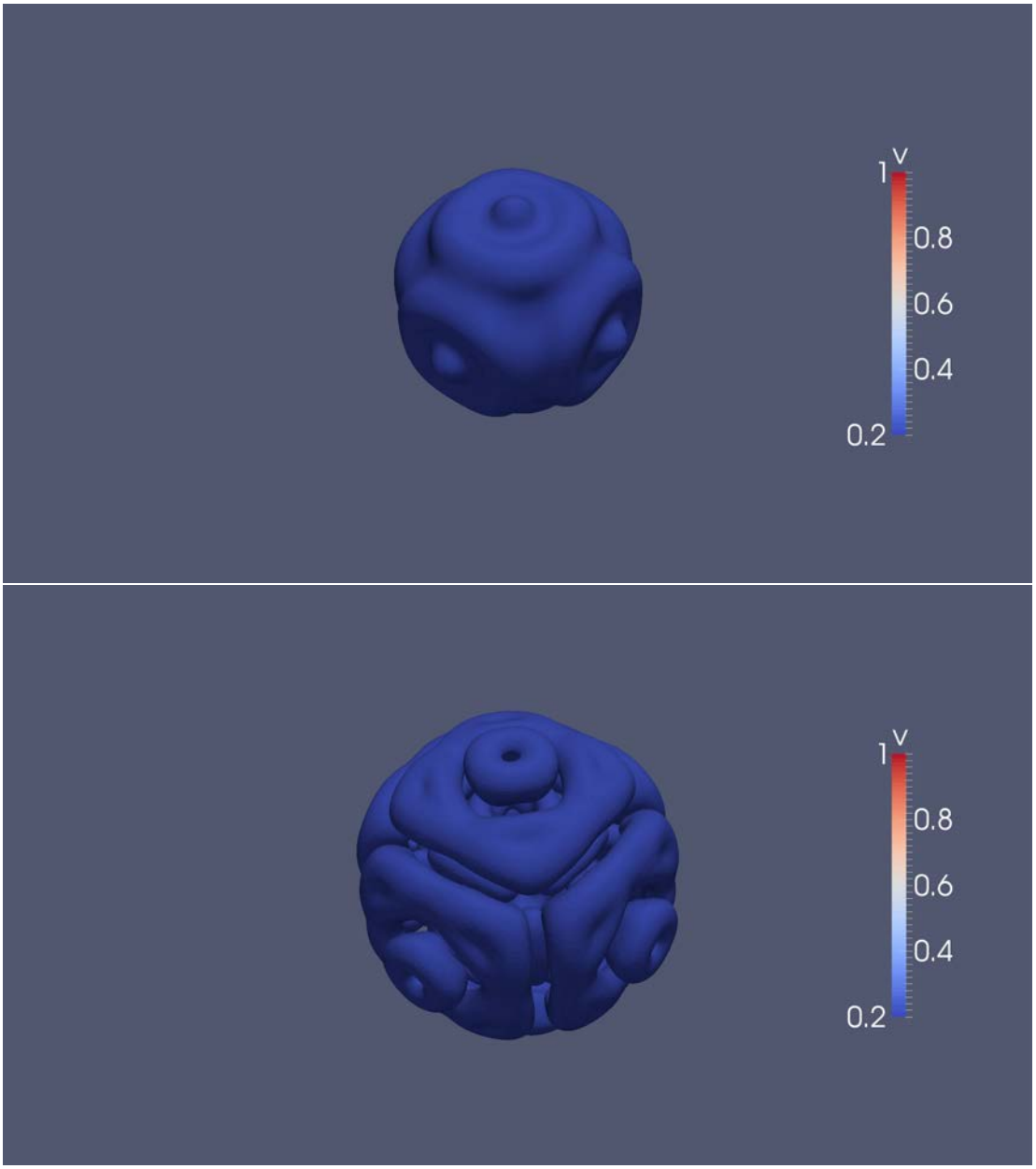
Um die Ergebnisse der Simulationen zu visualisieren wurde ParaView 4.1[6] genutzt. Diese Software ist ein multifunktionales Tool um Daten zu analysieren und visualisieren. Der ParaView-Client kann die Daten die von der Simulation abgespeichert wurden einlesen und in Echtzeit, für angemessene Problemgrößen, durch Anwenden verschiedener Filter optisch aufbereiten. Weiters beherrscht ParaView auch noch CoProcessing[1], das heißt, es kann während der Laufzeit der Simulation ausgegebenen Daten übernehmen, filtern und dann erst die Ergebnisse auf der Festplatte abspeichern. Die Arbeit an der Visualisierung begann im Client, mit einer Simulation mit geringer Gittergröße (64^3), da dort alle Parameter der Visualisierung, wie zum Beispiel die Auswahl der Filter, Farbschema, Betrachtungswinkel, ..., eingerichtet werden können. Dann konnten diese Parameter exportiert und im CoProcessing verwendet werden. Der Vorteil von Coprocessing ist das es den Umweg über die Festplatte spart, der durch den limitierten In-/Output zum Flaschenhals werden kann.

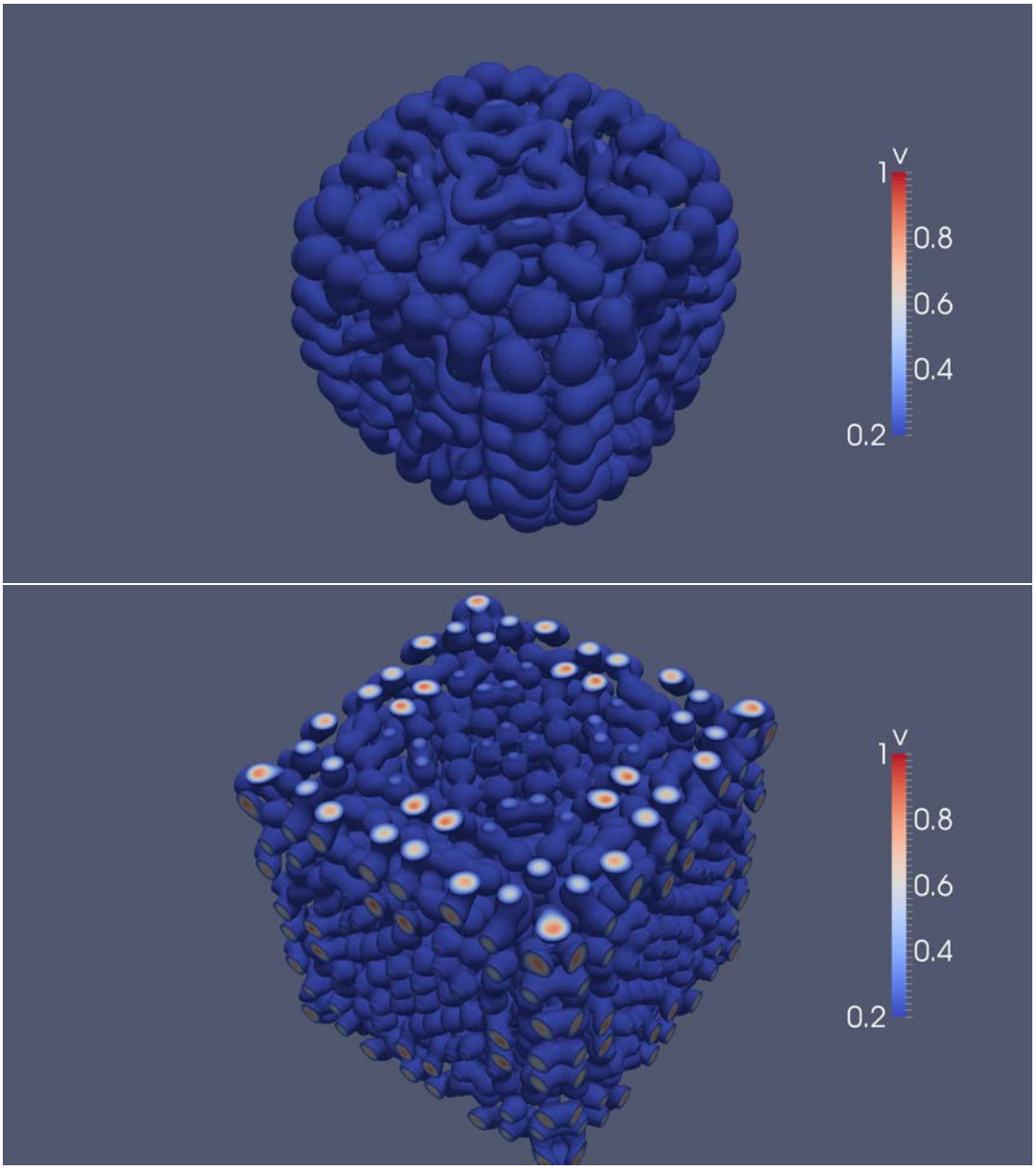
7.1 Bilder

Anfangs Bedingung der Komponente v und an $t = 125$. Es wurde ein Iso-Volume Filter angewandt.



Weiter Ergebnisse der Simulation von der Komponente v zu den Zeiten $t = 250, 300, 400, 500$. Es wurde der selbe Filter angewendet.


V
1
0.8
0.6
0.4
0.2
V
1
0.8
0.6
0.4
0.2



7.2 Videos

Es sind auch komplette Videos auf meiner Homepage verfügbar.
Homepage: `http://web.student.tuwien.ac.at/~e1226394/`
Direktlink: `https://www.youtube.com/watch?v=3CrbkSV2C60`

Literatur

[1] Nathan Fabian, Kenneth Moreland, David Thompson, Andrew C Bauer, Patrick Marion, Berk Geveci, Michel Rasquin, and Kenneth E Jansen. The paraview coprocessing library: A scalable, general purpose in situ visualization library. In *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*. IEEE, 2011.

[2] P. Gray and SK. Scott. Autocatalytic reactions in the isothermal, continuous stirred tank reactor: isolas and other forms of multistability. *Chemical Engineering Science*, 38, 1983.

[3] E. Hairer, S. Norsett, and G. Wanner. *Solving Ordinary Differential Equations I. Nonstiff Problems*. Springer, Berlin, 2nd rev. ed. 1993. corr. 3rd printing edition, 1993.

[4] O. Koch, Ch. Neuhauser, and M. Thalhammer. Embedded exponential operator splitting methods for the time integration of nonlinear evolution equations. *Applied Numerical Mathematics*, 63, 2013.

[5] N. Li and S. Laizet. 2DECOMP&FFT – A highly scalable 2D decomposition library and FFT interface. 2010.

[6] Amy Henderson Squillacote and James Ahrens. *The paraview guide*, volume 366. Kitware, 2007.

[7] W. Auzinger und O. Koch. Numerik von Differentialgleichungen, Vorlesungskriptum TU-Wien. 2014.

[8] `http://vsc.ac.at/`. Vienna Scientific Cluster.