

# Kinetik auf dem Rechner

---

## 1 Einleitung

Die chemische Reaktionskinetik wird bekanntlich durch Systeme von Differentialgleichungen (Dgl.) beherrscht, deren Lösungen (Integrale) Konzentrations-Zeit-Profile bezüglich der einzelnen Reaktanden liefern.

In einigen wenigen Fällen, etwa der Folgereaktion  $A \xrightarrow{k_1} B \xrightarrow{k_2} C$ , lassen sich *analytische* Lösungen des entsprechenden Systems von Dgl. finden, also *explizite* Lösungsfunktionen angeben.

Für komplexere Systeme mit einer Vielzahl von Reaktanden und Schritten ist dies normalerweise nicht mehr möglich. In der Vergangenheit wurden dann Näherungslösungen herangezogen, die auf Kosten der quantitativen Exaktheit ein qualitativ richtiges Bild von Reaktionsverläufen zeigen. Ein wichtiges Beispiel ist die Quasistationaritätsnäherung (vgl. Vorlesungen der Physikalischen Chemie).

Der vorliegende „Rechnerversuch“ soll zeigen, wie die Kinetik komplexer chemischer Systeme quantitativ beliebig exakt mittels *numerischer* Methoden gelöst werden kann.

Zur Lösung wird die Arbeitsumgebung IGOR PRO (*wavemetrics*) verwendet, die den Studierenden im Physikalisch-Chemischen Praktikum kostenlos zur Verfügung steht. Es empfiehlt sich, zum Praktikumstermin ein Laptop mit installiertem IGOR PRO mitzubringen. Es genügt ein Computer je Praktikantengruppe.

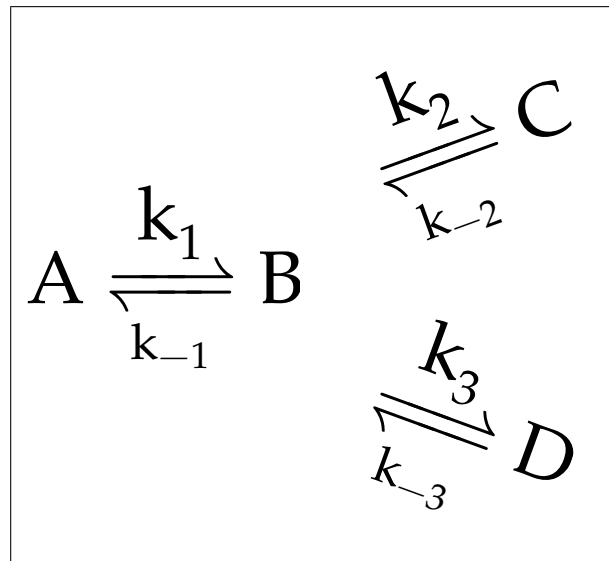


Abbildung 1: Schematische Darstellung eines Systems gekoppelter Reaktionen, das mittels numerischer Integration der entsprechenden gekoppelten Differentialgleichungen gelöst werden soll.

## 2 Beschreibung des zu untersuchenden reaktionskinetischen Schemas

Das zu untersuchende System wird in der Abb. 1 dargestellt.

In Worten: Komponente A geht in einer Gleichgewichtsreaktion mit den Geschwindigkeitskonstanten  $k_1$  und  $k_{-1}$  in die Substanz B über. B setzt sich in einer Folge-reaktion zu C um; diese Reaktion ist ebenfalls eine Gleichgewichtsreaktion mit der Geschwindigkeitskonstanten  $k_2$  und  $k_{-2}$ . Als Konkurrenzprozess tritt die Umsetzung von B zu D mit den Geschwindigkeitskonstanten  $k_3$  und  $k_{-3}$  hinzu. Das Gesamtsystem strebt also einem Gleichgewichtszustand zu, bei dem die Konzentrationen der Komponenten A–D von den Geschwindigkeitskonstanten abhängen.

Für die einzelnen Komponenten lauten die Dgl. wie folgt:

$$\frac{d[A]}{dt} = k_{-1} \cdot [B] - k_1 \cdot [A] \quad (1)$$

$$\frac{d[B]}{dt} = k_1 \cdot [A] - k_{-1}[B] - k_2 \cdot [B] + k_{-2} \cdot [C] - k_3 \cdot [B] + k_{-3} \cdot [D]. \quad (2)$$

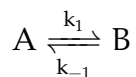
$$\frac{d[C]}{dt} = k_2 \cdot [B] - k_{-2} \cdot [C] \quad (3)$$

$$\frac{d[D]}{dt} = k_3 \cdot [B] - k_{-3} \cdot [D] \quad (4)$$

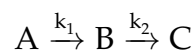
Eckige Klammern stehen für die Konzentrationen.

Aus dem oben angegebenen Schema können wir durch „Abschalten“ (Null setzen) von Geschwindigkeitskonstanten einfachere Prozesse erzeugen. Dies wird im Folgenden an einigen Beispielen gezeigt:

1. *Reversible Reaktion erster Ordnung.*– Es liegt ein chemisches Gleichgewicht zwischen A und B vor, wenn wir  $k_2$ ,  $k_{-2}$ ,  $k_3$  und  $k_{-3}$  abschalten:



2. *Folgereaktion ohne Gleichgewicht:* Wir können alle Rückreaktionen und zusätzlich den Prozess  $B \rightleftharpoons D$  abschalten, indem wir die entsprechenden Geschwindigkeitskonstanten jeweils gleich Null setzen. Die Reaktion sieht dann wie folgt aus:



## 3 Numerische Lösung von Differentialgleichungen mit IGOR PRO

Die Software IGOR PRO (wavemetrics) bietet eine in das Programm integrierte Möglichkeit, Differentialgleichungen erster Ordnung und auch Systeme gekoppelter Differentialgleichungen erster Ordnung numerisch zu lösen, also die *Zahlenwerte* der Lösungsfunktion zu erzeugen und in eine *wave* zu schreiben, ohne dass die Lösungsfunktion explizit genutzt wird.

Der Name der zu nutzenden Routine lautet `integrateODE`.

Zur Nutzung von `integrateODE` wird vom Anwender eine nutzerdefinierte Funk-

## Kinetik auf dem Rechner

---

tion festgelegt, die die Differentialgleichung(en) enthält. Außerdem muss eine *Anfangsbedingung* gesetzt werden, um aus der Lösungsschar der Dgl. eine spezielle Lösung auszuwählen.

Der einfachste Aufruf von `integrateODE` lautet:

```
integrateODE derivFunc, paramWave, resultWave
```

Es werden drei Größen an `integrateODE` übergeben:

1. `derivFunc`: die vom Anwender aufzustellende Differentialgleichung bzw. das System von Differentialgleichungen;
2. `paramWave`: Name einer *wave*, die Koeffizienten (Geschwindigkeitskonstanten) enthält, die an `derivFunc` übergeben werden müssen;
3. `resultWave`: Name der *Wave*, in die die Ergebnisse der numerischen Integration von IGOR eingeschrieben werden; also derjenigen *Wave*, die das eigentliche Resultat von `integrateODE` enthält. Falls die Konzentration mehrerer Reaktanden verfolgt werden soll, muss die entsprechende *Wave* mehrspaltig sein (eine Spalte je Reaktand).

Weitere Parameter *können* an `integrateODE` übergeben werden, um die Vorgehensweise näher zu spezifizieren. Für diesen Praktikumsversuch sind aber keine weiteren Parameter notwendig.

Die eigentliche Arbeit des Anwenders besteht darin, `derivFunc` entsprechend der Problemstellung aufzustellen (zu programmieren).

Die Funktion `derivFunc` hat stets die folgende Form:

```
Function Dgl(pw, xx, yw, dydx)
Wave pw
Variable xx
Wave yw
Wave dydx

dydx[0] = <expression for one derivative>
dydx[1] = <expression for next derivative>
<etc.>

return 0
End
```

*Erklärungen:* Die Funktion `Dgl` wird nicht vom Anwender aufgerufen, sondern die in IGOR bereits fertig vorhandene Routine `integrateODE` ruft `Dgl` auf. `pw` ist die der Funktion `Dgl` eigene interne Bezeichnung für die an `Dgl` übergebene Parameterwave (`paramWave`), deren Namen Sie beim Aufruf von `integrateODE` an diese Routine übergeben; `integrateODE` transferiert die Wave dann an die Funktion `Dgl` (wo sie als Argument `pw` auftritt). `xx` benötigen Sie nicht, Sie müssen es aber dennoch deklarieren. `yw` ist eine von `integrateODE` intern erzeugte temporäre Wave, um die Sie sich ebenfalls nicht zu kümmern brauchen, da ihre Verwaltung IGOR obliegt. `dydx` ist ebenfalls eine von IGOR intern verwaltete Wave, deren Elemente die einzelnen Differentialgleichungen sind, die Sie lösen wollen.

`dydx` ist das entscheidende Element der `function`. Hier tragen Sie die gekoppelten Dgl. ein. Falls nur *eine* Dgl. vorliegt, enthält `dydx` nur *ein* Element `dydx[0]`. Falls ein Satz gekoppelter Dgl. vorliegt, enthält `dydx` sovielen Elemente, wie gekoppelte Dgl. vorliegen (siehe Beispiel weiter unten), d.h.. `dydx[i]` enthält die *i*-te Dgl.

### 3.1 Beispiel

Die Dgl.

$$\frac{dy(x)}{dx} = -k \cdot y(x),$$

beschreibt irreversible Reaktionen erster Ordnung.  $y(x)$  steht für die Konzentration des Eduktes,  $x$  ist die unabhängige Variable (die Zeit). Die Lösung lautet bekanntlich

$$y(x) = C \cdot \exp(-k \cdot x)$$

mit der beliebig wählbaren Integrationskonstante  $C$ . Aus der Lösungsmenge lässt sich eine *spezielle* Lösung auswählen, indem die Konstante  $C$  als Randbedingung festgelegt wird. Dies erfolgt durch Festlegen des Punktes  $y(x = 0)$  (*Anfangswert*). Dieses Beispiel einer Dgl. lässt sich natürlich ganz einfach analytisch integrieren, aber wir sind hier an numerischen Lösungen interessiert.

Im Falle der genannten Dgl. muss die Parameter-Wave `paramWave` genau *ein* Element enthalten, nämlich den Zahlenwert von  $k$ .

Wir definieren nun die vom Anwender festzulegende `derivFunc`, der wir in diesem Beispiel den Namen `Dgl1` geben:

```
function Dgl1(pw, xx, yw, dydx)
    wave pw
    variable xx
    wave yw
```

```
    wave dydx
    dydx[0] = -pw[0]*yw[0]
    return 0
end
```

Die function `Dg11` können Sie aus dieser Skripte in das „procedure window“ von IGOR kopieren. (Sie können das Procedure Window öffnen, indem Sie in der Igor-Menü-Leiste **Windows** → **Procedure Windows** → **Procedure Window** anklicken.)

Nun lassen wir die in `dydx[0]` definierte Differentialgleichung durch `integrateODE` numerisch lösen. Die Resultate soll `integrateODE` in eine Wave (die `resultWave`) schreiben, der wir willkürlich den Namen `resultDg11` geben. Der  $x$ -Bereich von `resultDg11` (in IGOR heißt die unabhängige Variable immer  $x$ ) soll auf den Bereich 0–20 skaliert werden. Wir müssen den Anfangswert von `resultDg11` festlegen, wir wählen willkürlich den Wert 10. Wir müssen ferner die Konstante  $k$  in `paramWave` festlegen; diese Wave nennen wir `paramWave1`. Wir wählen willkürlich die Geschwindigkeitskonstante zu  $k = 0,5$ . Hierzu müssen insgesamt die folgenden Kommandos an der IGOR-Kommandozeile eingegeben werden:

```
make/o/N=256 resultDg11
setscale x, 0,20,resultDg11
resultDg11[0]=10
display resultDg11
make/o/N=1 paramWave1
paramWave1[0]=0.5
```

Nun können wir die Differentialgleichung numerisch integrieren und die Resultate graphisch beobachten. Wir geben an der Kommandozeile ein:

```
integrateODE Dg11, paramWave1, resultDg11
```

Das Ergebnis sollte wie in Abb. 2 aussehen. Wir erkennen, dass die Zahlenwerte auf einer fallenden Exponentialfunktion liegen. Eine logarithmische Darstellung der Ordinate zeigt dementsprechend eine Gerade als Funktion der Zeit. Dies ist in der Abb. 3 dargestellt.

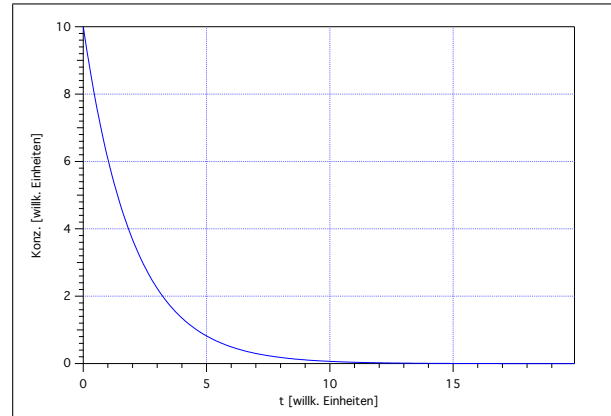


Abbildung 2: Numerische Lösung der Differentialgleichung Dgl1 mit Hilfe von `integrateODE`.

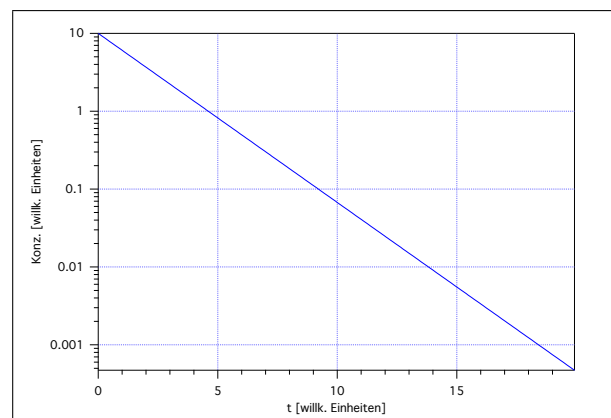
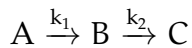


Abbildung 3: Logarithmische Darstellung der numerischen Lösung der Differentialgleichung Dgl1 aus Abb 2.

## 3.2 System gekoppelter Differentialgleichungen

Wir betrachten die Folgereaktion:



Es liegen drei Dgl. vor, die simultan erfüllt sein müssen:

1. 
$$\frac{d[A]}{dt} = -k_1 \cdot [A]$$

2. 
$$\frac{d[B]}{dt} = k_1 \cdot [A] - k_2 \cdot [B]$$

3. 
$$\frac{d[C]}{dt} = k_2 \cdot [B]$$

Wir wollen die folgende Anfangssituation annehmen:

$$[A]_0 = 1, [B]_0 = 0, [C]_0 = 0.$$

(Wir verzichten hier auf die Angabe der Einheiten der Konzentration.)

Wir müssen folgende Modifikationen an der bisherigen Vorgehensweise vornehmen:

1. Wir brauchen eine `paramWave` mit *zwei* Elementen (entsprechend der *zwei* Konstanten  $k_1$  und  $k_2$ );
2. Wir benötigen eine *dreispaltige* Lösungswave zur Aufnahme der Konzentrationen von A, B und C.
3. Da insgesamt *drei* Dgl. vorliegen, benötigen wir Ausdrücke `dydx[0]`, `dydx[1]` und `dydx[2]`; die Ausdrücke entnehmen wir dem obigen Dgl.-System.

Die abgewandelte `derivFunc` nennen wir `Dgl3`:

```
function Dgl3(pw, xx, yw, dydx)
    wave pw
    variable xx
    wave yw
```



## Kinetik auf dem Rechner

---

```
    wave dydx
    dydx[0] = -pw[0]*yw[0]
    dydx[1] = pw[0]*yw[0] - pw[1]*yw[1]
    dydx[2] = pw[1]*yw[1]
    return 0
end
```

*Hinweis:* yw[0] und yw[1] sind temporär von integrateODE erzeugte Waves, die Zahlenwerte der Konzentrationen von A und B enthalten.

Die paramWave nennen wir paramWave3 und wir legen sie wie folgt fest:

```
make/o/N=2 paramWave3
paramWave3[0]=1
paramWave3[1]=1
```

Wir arbeiten also mit gleichen Koeffizienten  $k_1 = 1$  und  $k_2 = 1$ .

Die resultWave wird nunmehr zweidimensional und enthält drei Spalten, entsprechend den Produkten A, B und C:

```
Make/O/N=(100,3) resultDgl3
```

Dies erzeugt die Wave resultDgl3 mit drei Spalten zu 100 Punkten.

Wir skalieren die Wave resultDgl3 auf den x-Bereich 0-10 und setzen die Anfangsbedingungen ein:

```
setscale x,0,10,resultDgl3
resultDgl3[0][0]=1
resultDgl3[0][1]=0
resultDgl3[0][2]=0
```

Wir stellen die Spalten von resultDgl3 graphisch in *einer* Abbildung dar:

```
display resultDgl3[][0]
appendtograph resultDgl3[][1]
appendtograph resultDgl3[][2]
```

Jetzt wird `integrateODE` aufgerufen und die aktuellen Argumente werden übergeben:

```
integrateODE Dgl3, paramWave3, resultDgl3
```

Das Resultat ist in der Abb. 4 abgebildet. Zur Verdeutlichung sind die einzelnen Spalten von `resultDgl3` in unterschiedlichen Farben dargestellt.

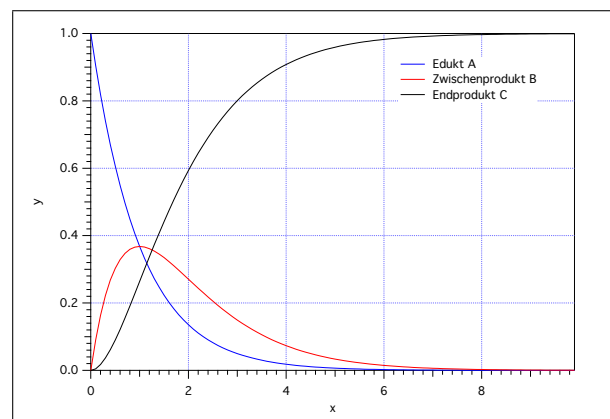


Abbildung 4: Numerische Lösung des Differentialgleichungssystems für das reaktionskinetische System  $A \xrightarrow{k_1} B \xrightarrow{k_2} C$ . Die Achsbezeichnungen sind willkürlich.

## 4 Praktische Durchführung des Versuches

Während des Praktikumstermines wird schrittweise unter Anleitung des Assistenten die Implementierung des Gleichungssystems entsprechend Abb. 1 erarbeitet.

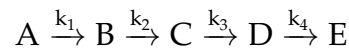
## 5 Aufgaben

1. Laden Sie unter der folgenden URL eine graphische Oberfläche zur Lösung des in Abb. 1 gezeigten Reaktionsschemas herunter:

<http://pcprakt.userpage.fu-berlin.de/KinetikODE.ipf>

Laden Sie das Modul durch Doppelklicken in IGOR ein. Klicken Sie auf „compile“. Klicken Sie im Menü **Macros** den Menüpunkt **Reaktionskinetik**. Setzen Sie alle Geschwindigkeitskonstanten des in der Abb. 1 dargestellten Reaktionssystems gleich Eins. Nach welcher Zeit ist das System im Gleichgewicht (Abweichungen von der Gleichgewichtskonzentration für alle Komponenten  $< 10^{-3}$  %)?

2. Bilden Sie das folgende Reaktionssystem in einer geeigneten `derivFunc` mit dem Namen `Dg1X1` ab:



Erzeugen Sie die Parameter-Wave `pwX1` mit den Geschwindigkeitskonstanten  $k_1 = 1$ ,  $k_2 = 0.1$ ,  $k_3 = 5$ ,  $k_4 = 1$ . Dokumentieren Sie in Ihrem Protokoll `Dg1X1` und `pwX1`. Als Randbedingung gelte:  $[A]_0 = 1$ . Integrieren Sie mittels `integrateODE`. Erstellen Sie eine Graphik, die das Konzentrations-Zeit-Profil aller Reaktanden in ansprechender Form enthält (Qualität der Graphik ist bewertungsrelevant).

3. Beurteilen Sie diese Skripte.
- Enthält die Skripte Fehler?
  - Ist es möglich, die Skripte zu kürzen?
  - Fehlen wichtige Informationen?